Putting AI on a Diet: TinyML and Efficient Deep Learning

Song Han

Massachusetts Institute of Technology

TinyML and Efficient Deep Learning

- **Optimize the Computation Efficiency**
  - Inference: MCUNet for IoT Devices [NeurIPS'20, spotlight]
  - Training: Tiny On-Device Transfer Learning (TinyTL) [NeurIPS'20]
- **Optimize the Data Efficiency**
  - Differentiable Augmentation for Data-Efficient GAN Training [NeurIPS'20]

TinyML and Efficient Deep Learning

- **Optimize the Computation Efficiency**
  - Inference: MCUNet for IoT Devices [NeurIPS'20, spotlight]
  - Training: Tiny On-Device Transfer Learning (TinyTL) [NeurIPS'20]
- **Optimize the Data Efficiency**
  - Differentiable Augmentation for Data-Efficient GAN Training [NeurIPS'20]

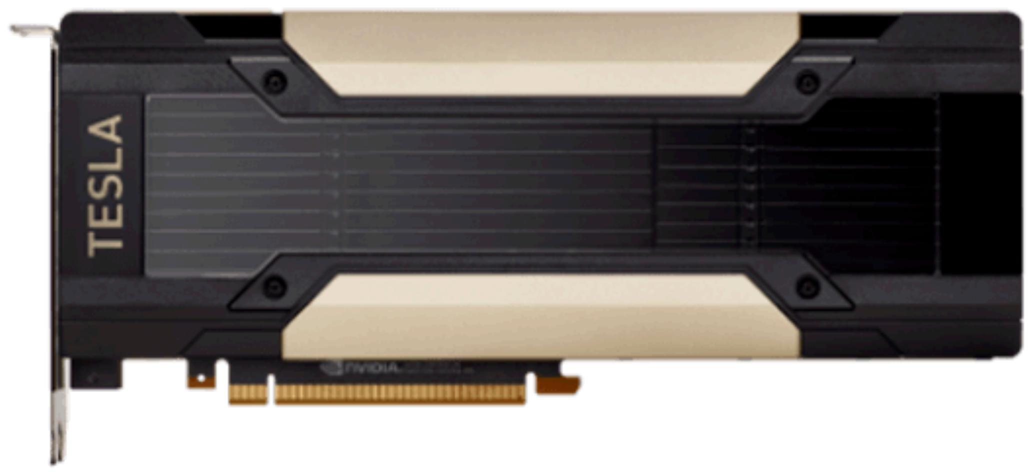
MCUNet: Tiny Deep Learning on IoT Devices

Ji Lin[1] Wei-Ming Chen[1,2] Yujun Lin[1] John Cohn[3] Chuang Gan[3] Song Han[1]

[1]MIT [2]National Taiwan University [3]MIT-IBM Watson AI Lab

NeurIPS 2020 (spotlight)

Challenge: Memory Too Small to Hold DNN



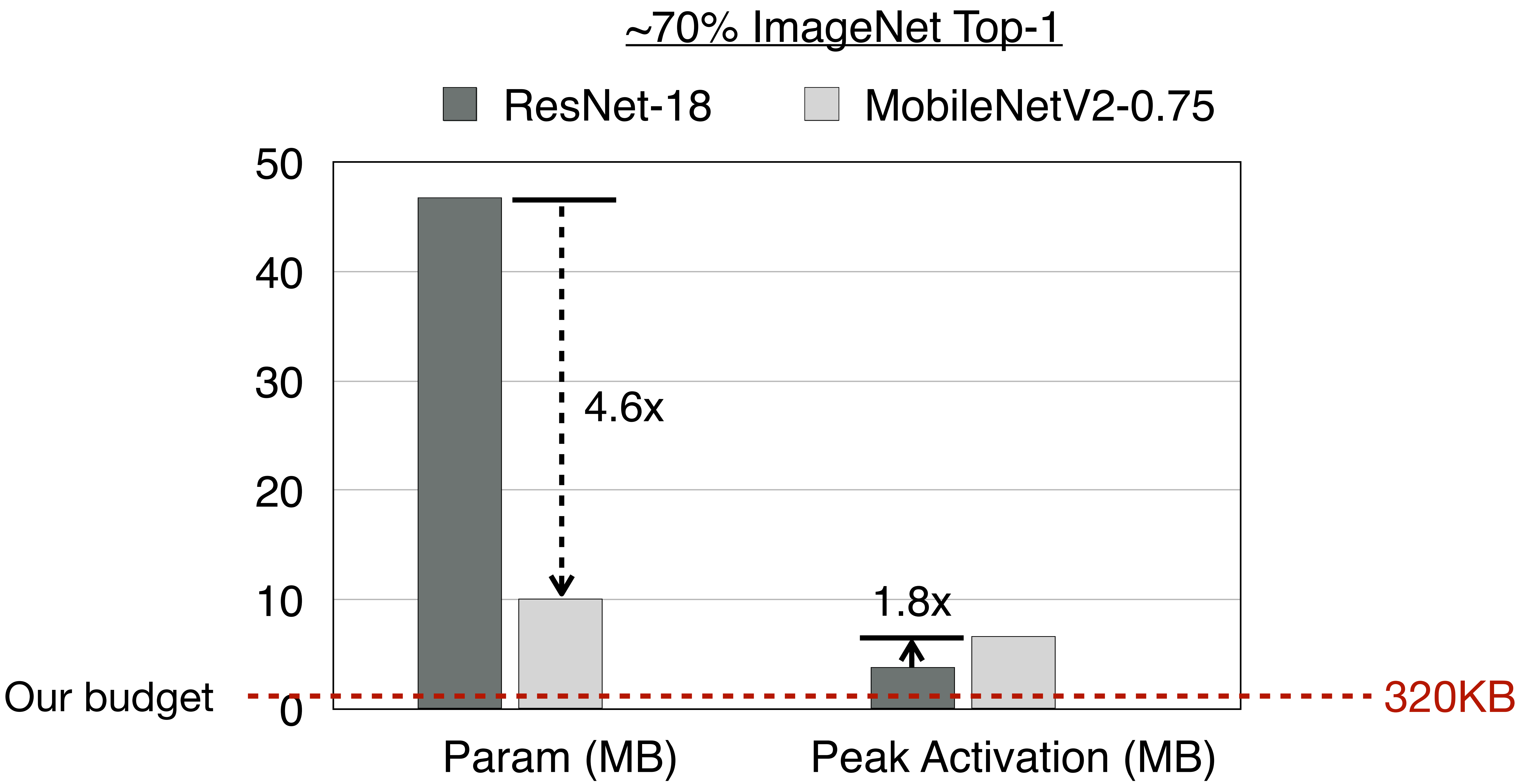
	Cloud AI	Mobile AI	Tiny AI
Memory (Activation)	16GB	4GB	320kB
Storage (Weights)	~TB/PB	256GB	1MB

13,000x smaller

50,000x smaller

We need to reduce the <u>peak activation size</u> **AND** the <u>model size</u> to fit a DNN into MCUs.

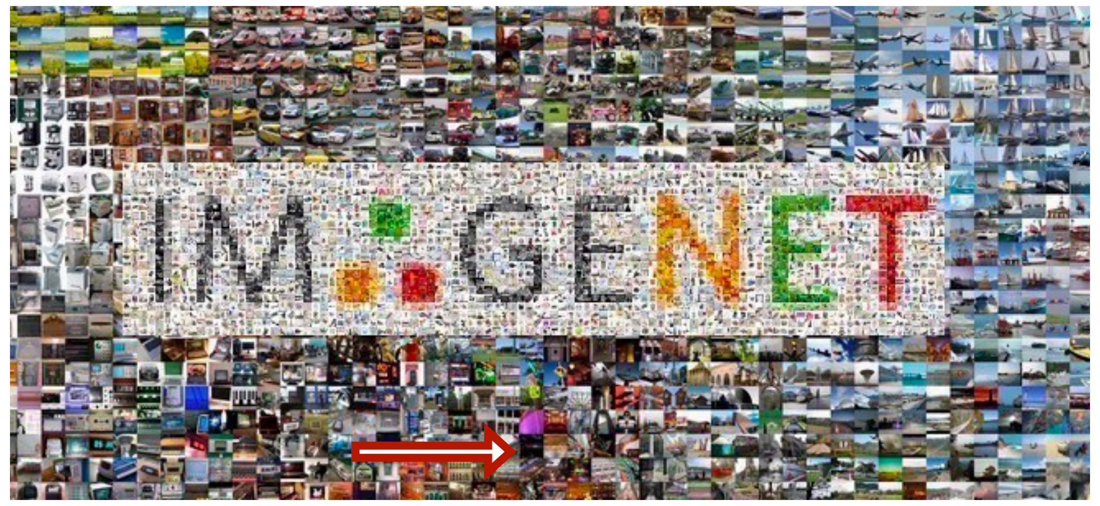
Existing efficient network only reduces model size but NOT activation size!



Tiny AI

Toy applications

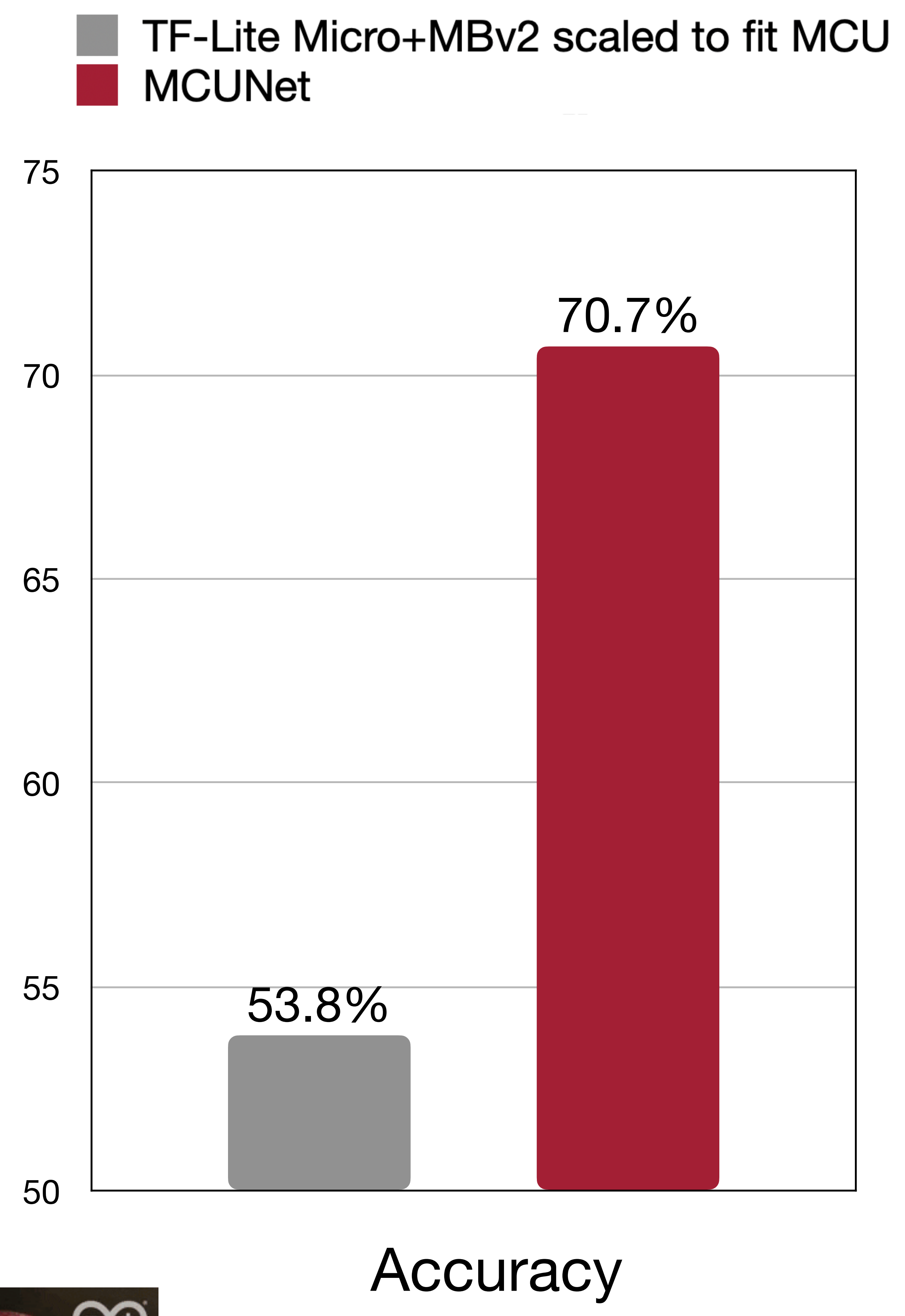
MCUNet



TinyML: Bring AI to IoT Devices

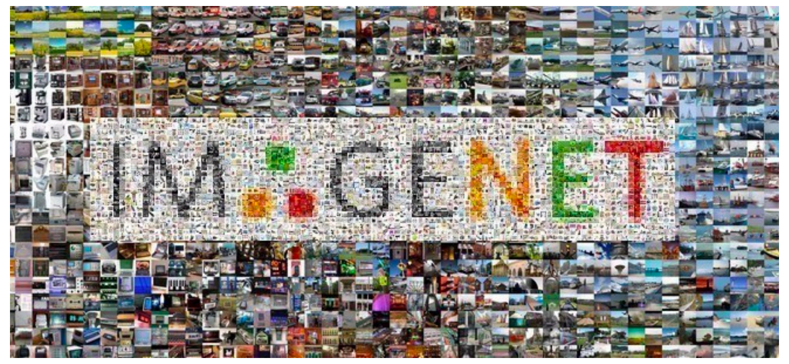
Challenging memory resource: 256KB SRAM, 1MB Flash on MCU
Key: co-design the neural architecture, the compiler and inference engine

MIT researchers have developed a system, called MCUNet, that brings machine learning to microcontrollers. The advance could enhance the function and security of devices connected to the Internet of Things (IoT). ——MIT News



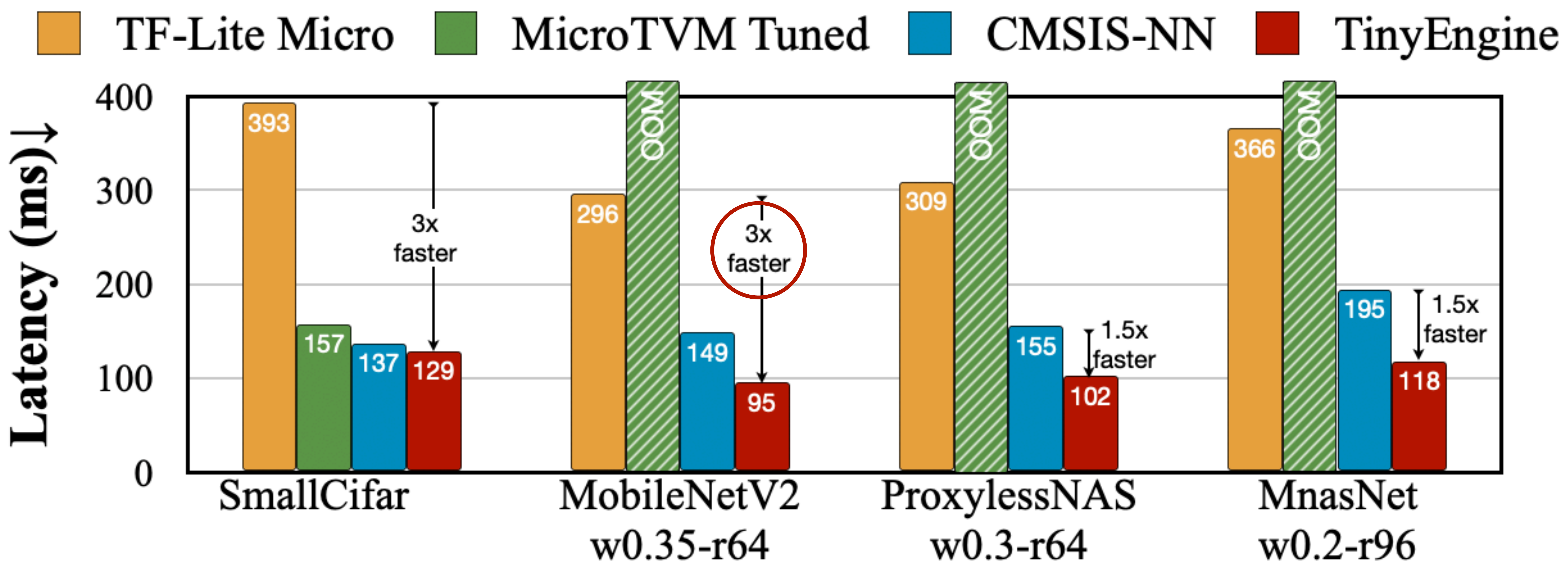
toy applications

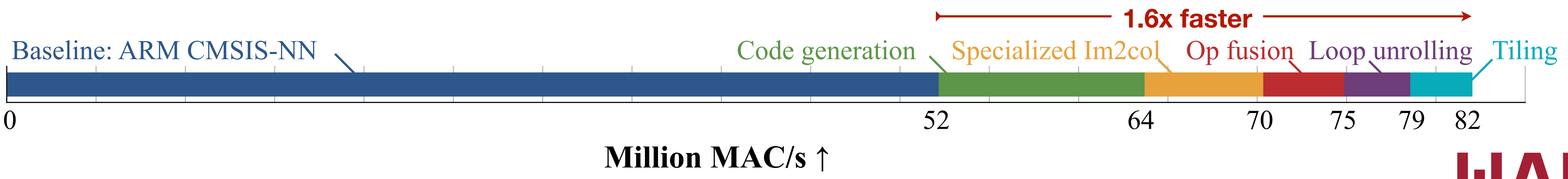
MCUNet



large scale

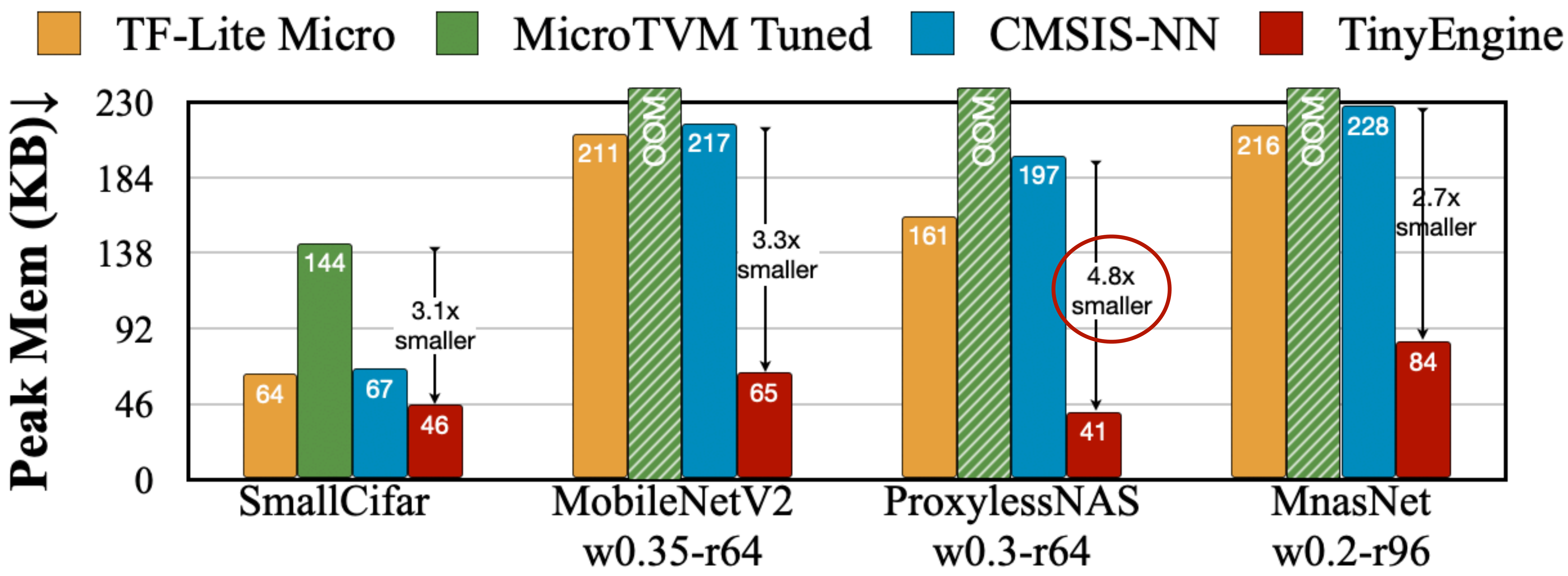
TinyEngine: Speedup





Measured on the same STM32 MCU

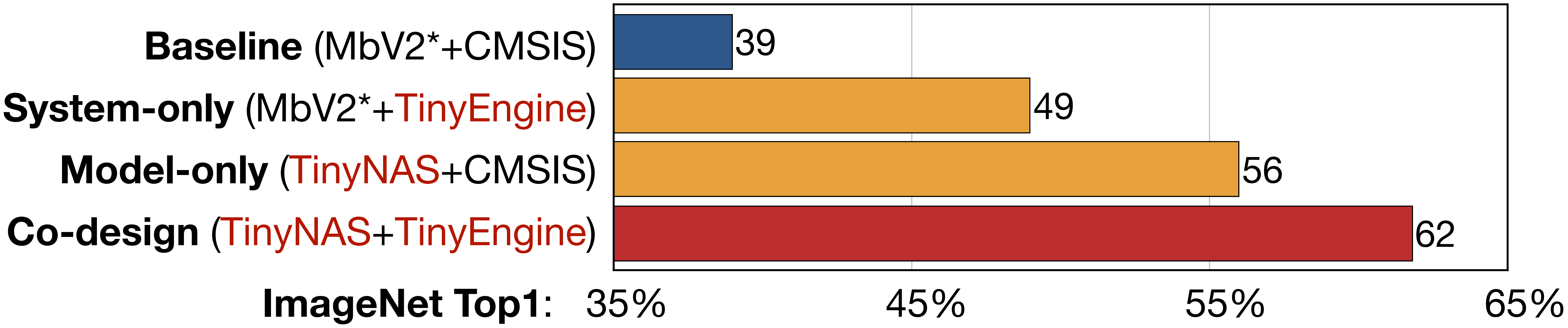
TinyEngine: Memory Saving



Measured on the same STM32 MCU

MCUNet: TinyNAS+TinyEngine

- ImageNet classification on STM32F746 MCU (**320kB SRAM**, **1MB Flash**)

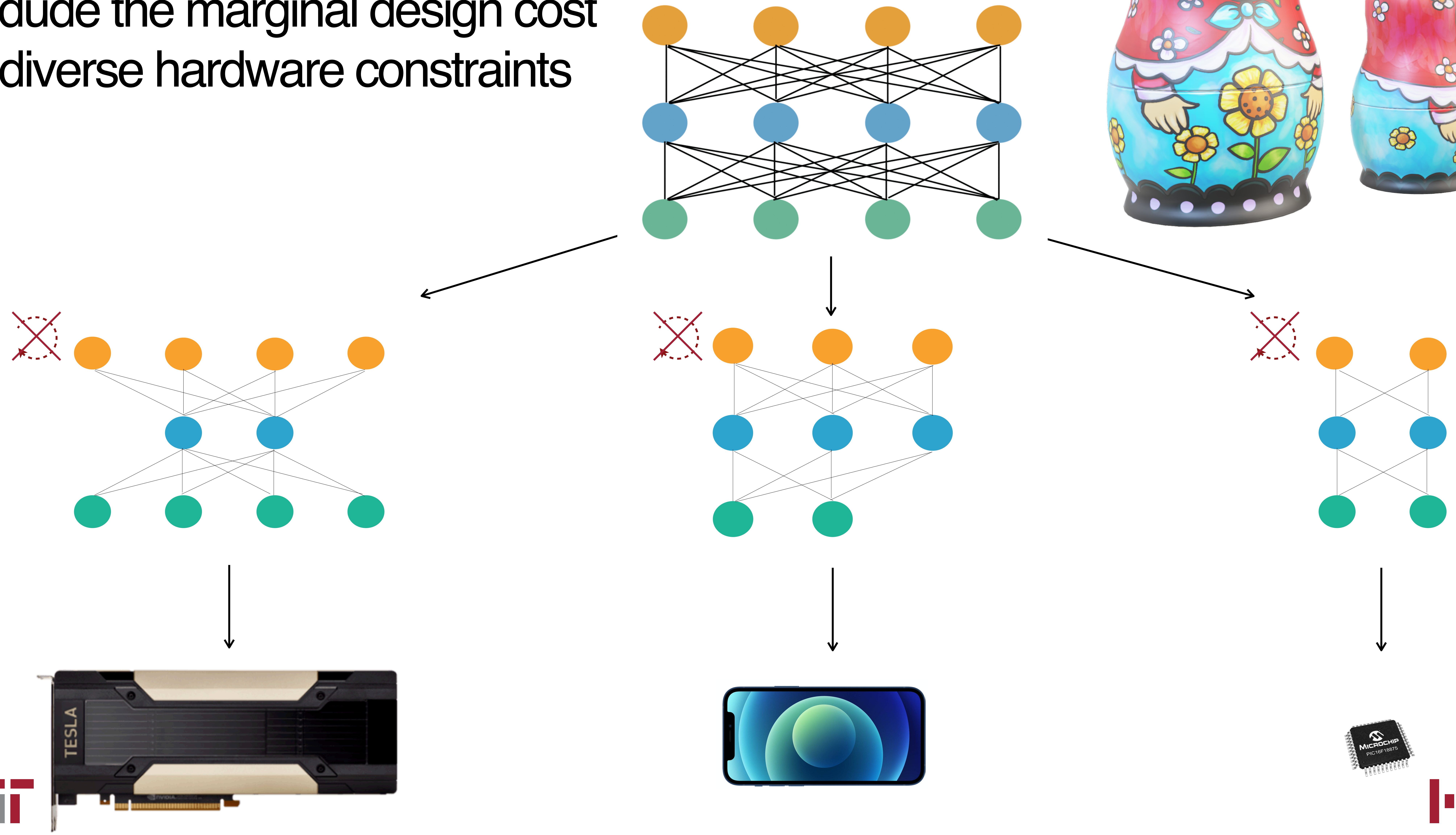


* scaled down version

Once-for-All Network

Train once, get many
Redude the marginal design cost
Fit diverse hardware constraints

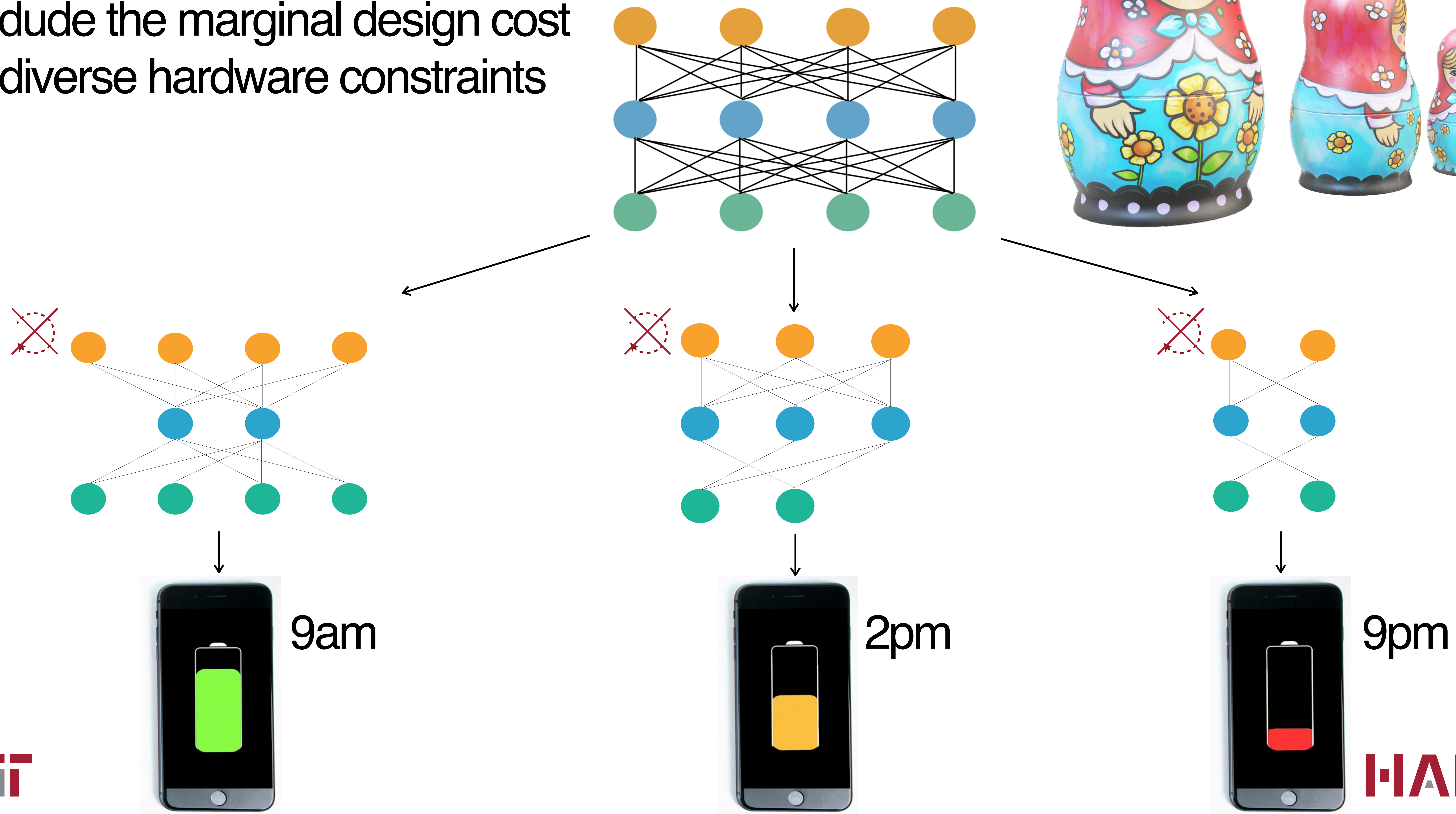
Smaller child networks are nested in larger ones



Once-for-All Network

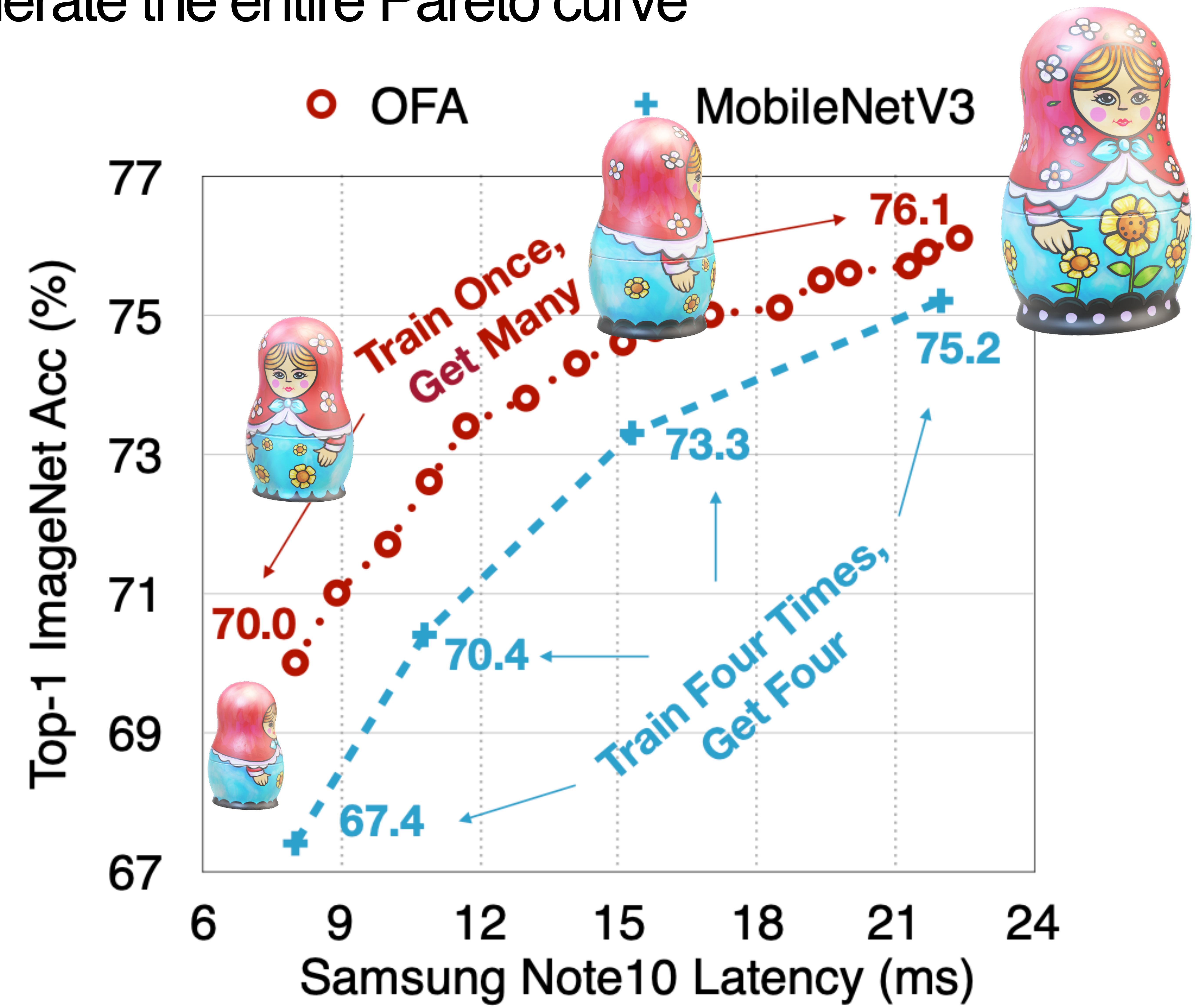
Train once, get many
Redude the marginal design cost
Fit diverse hardware constraints

Smaller child networks are nested in larger ones



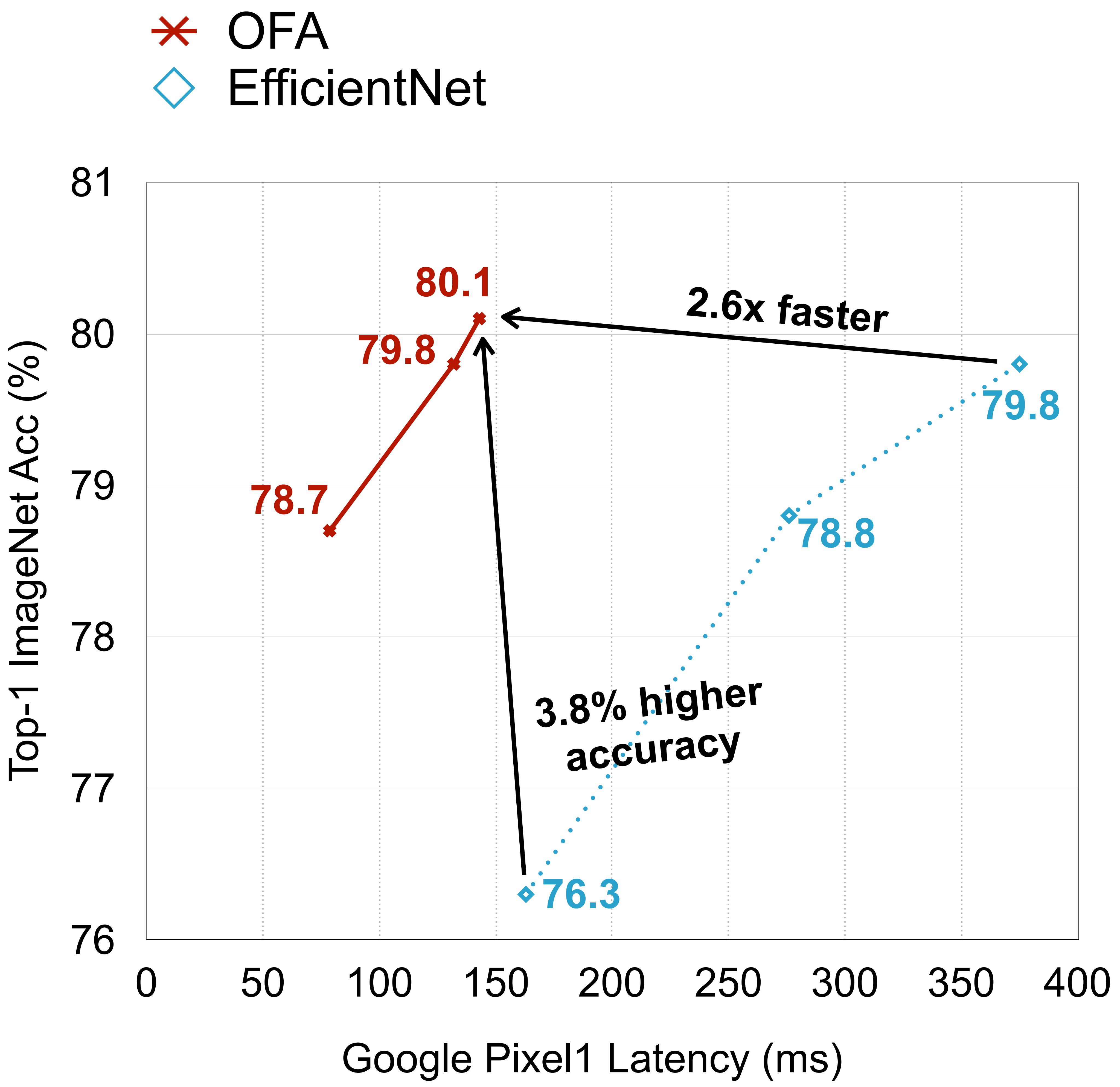
Once-for-All Network

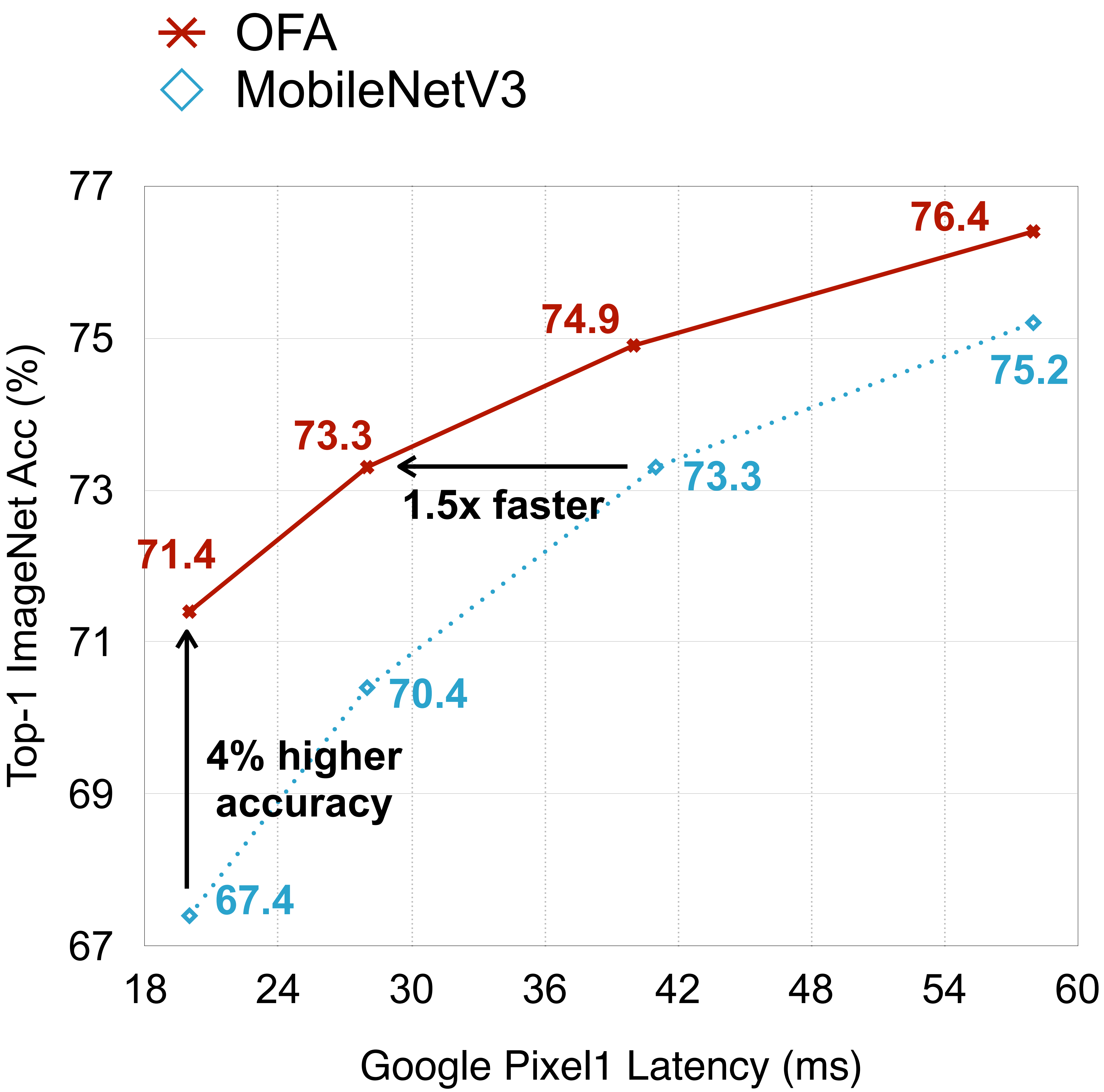
Train only once, generate the entire Pareto curve



Once-for-All Network

Train only once, handle diverse hardware constraints

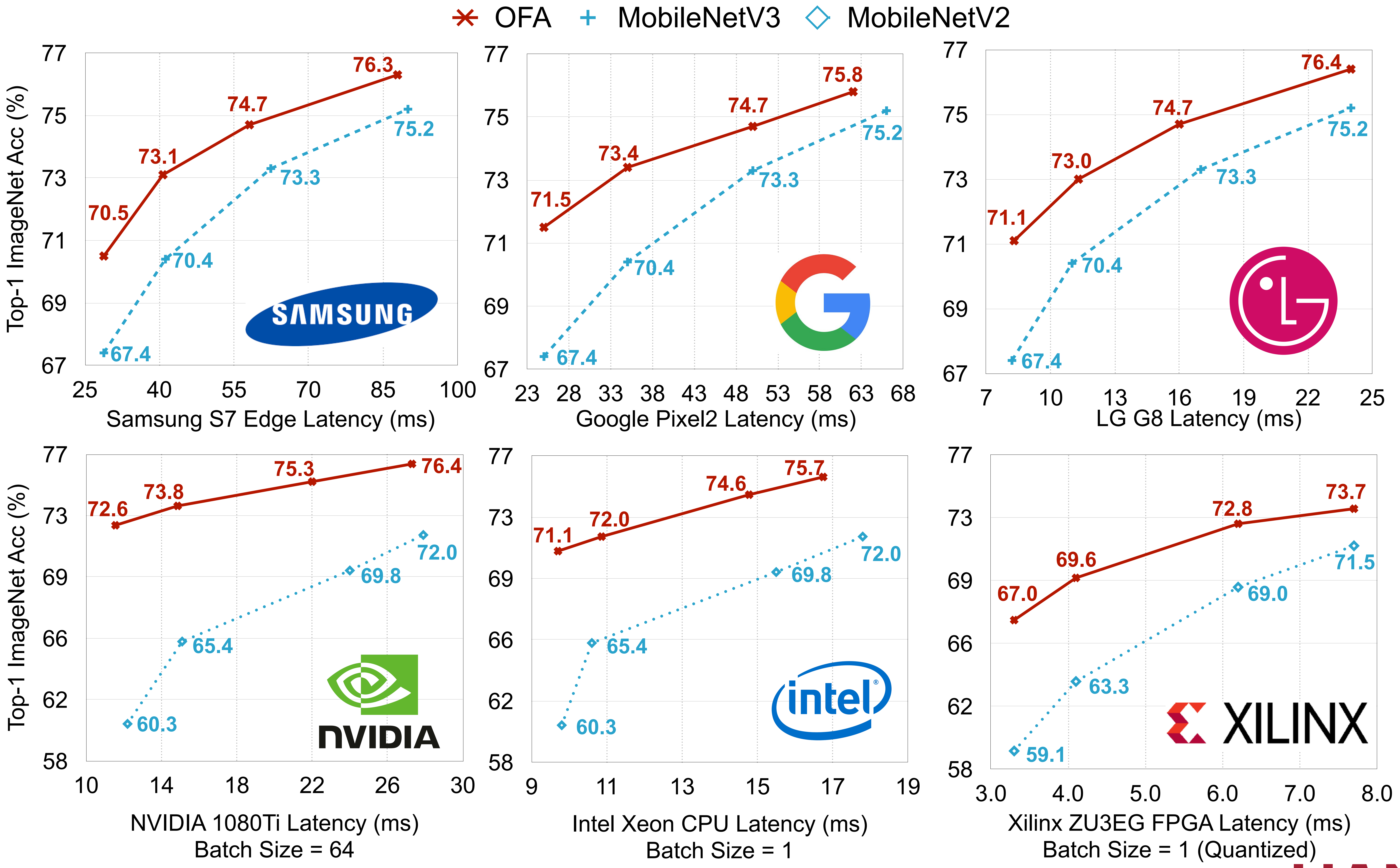




- Training from scratch cannot achieve the same level of accuracy

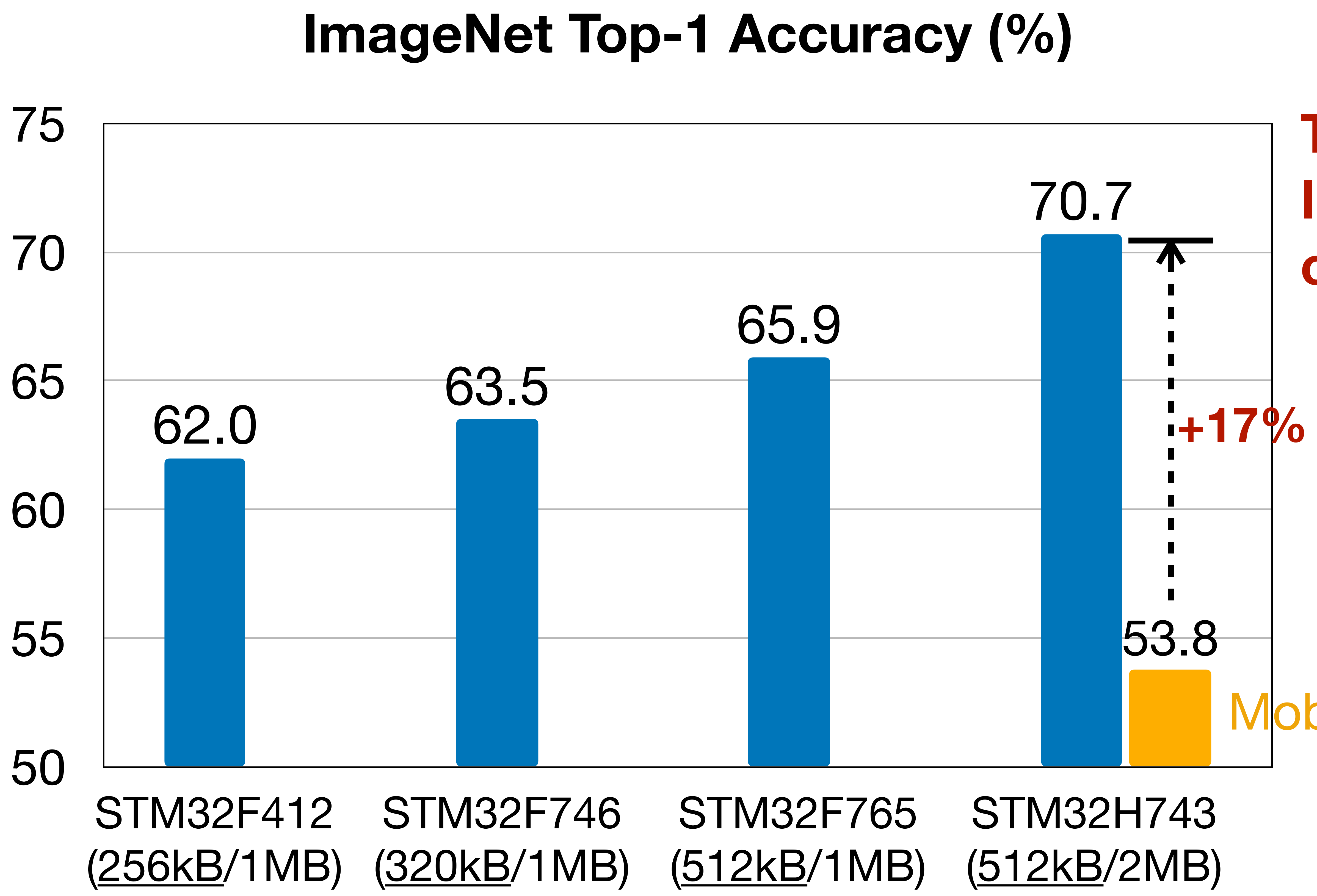
Once-for-All, ICLR'20

Once-for-All Network



Once-for-All Network

- Specializing models (int4) for different MCUs (SRAM/Flash)

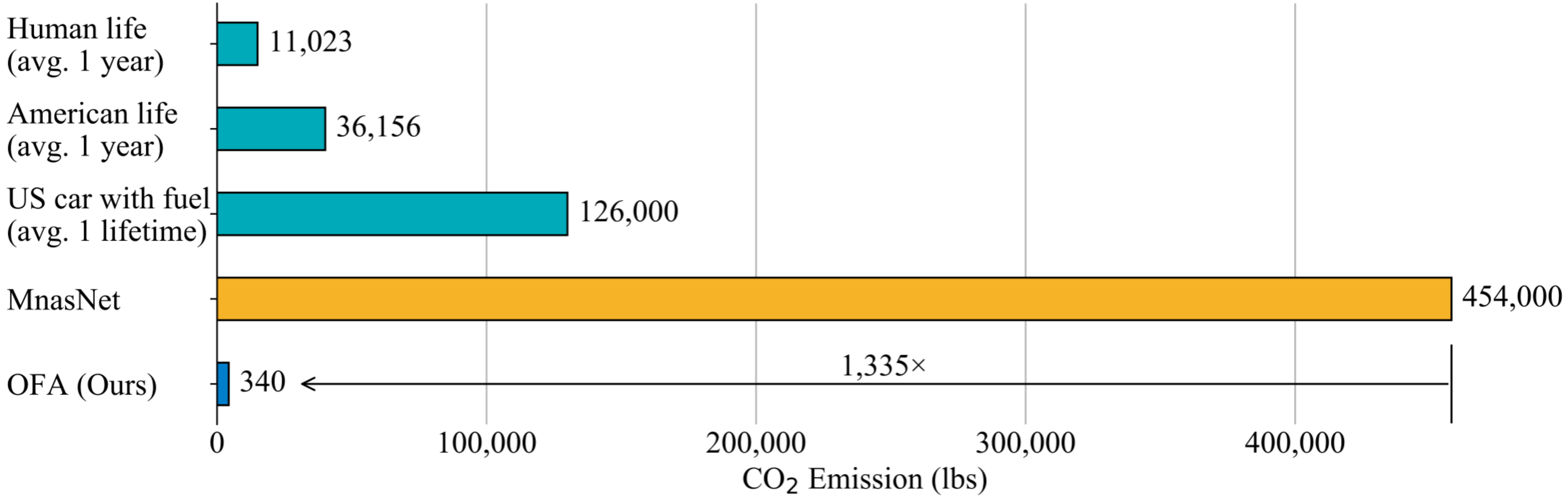


The first to achieve >70% ImageNet accuracy on commercial MCUs

Once-for-All Network

Consistently Outperforms Human Baselines

Turn-key solution for many hardware platforms: CPU/GPU/DSP/FPGA

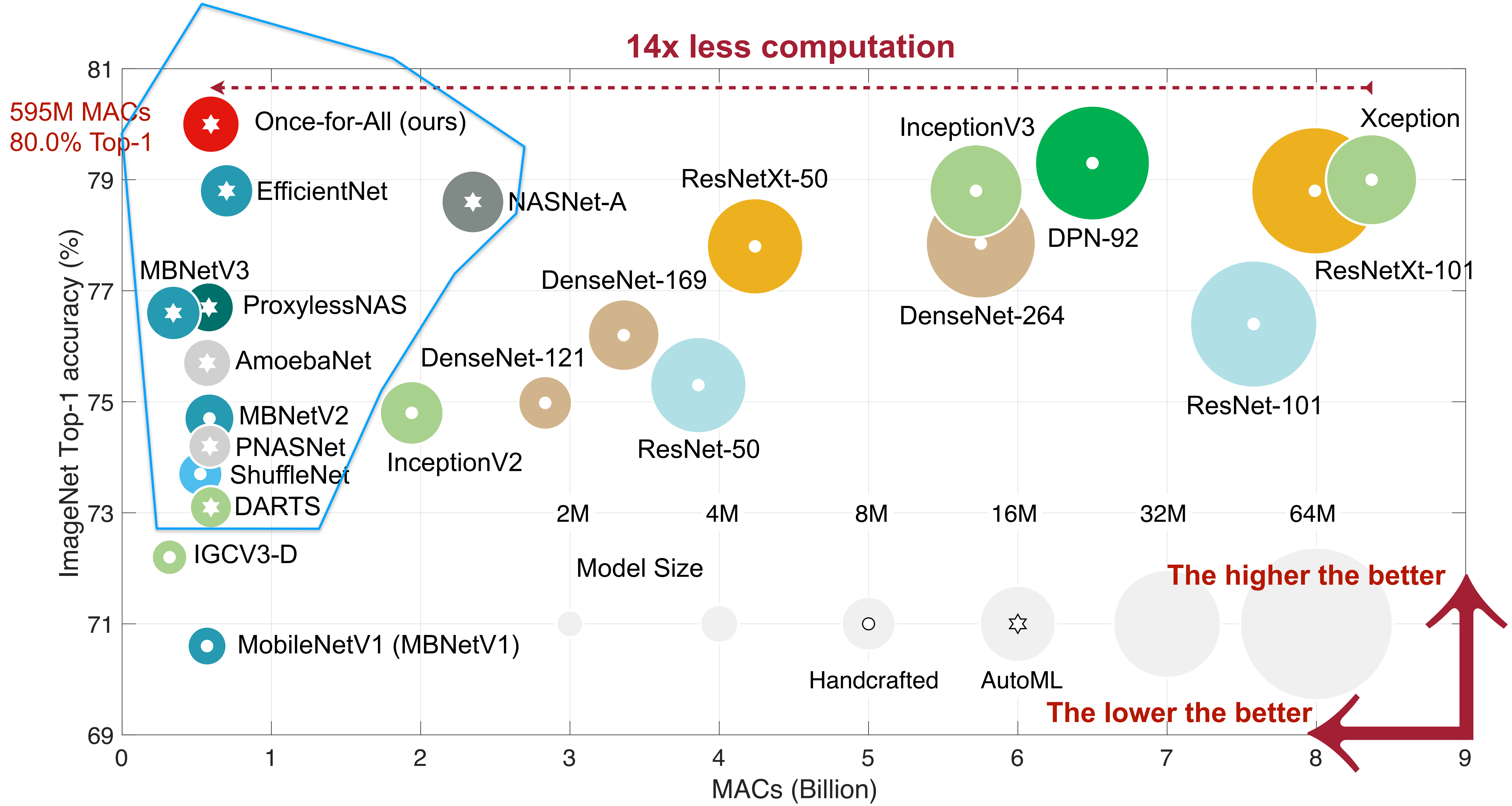


- **Six first-place finishes in top competitions in efficient AI**

AutoML, Neural Architecture Search

Consistently outperforms human baselines

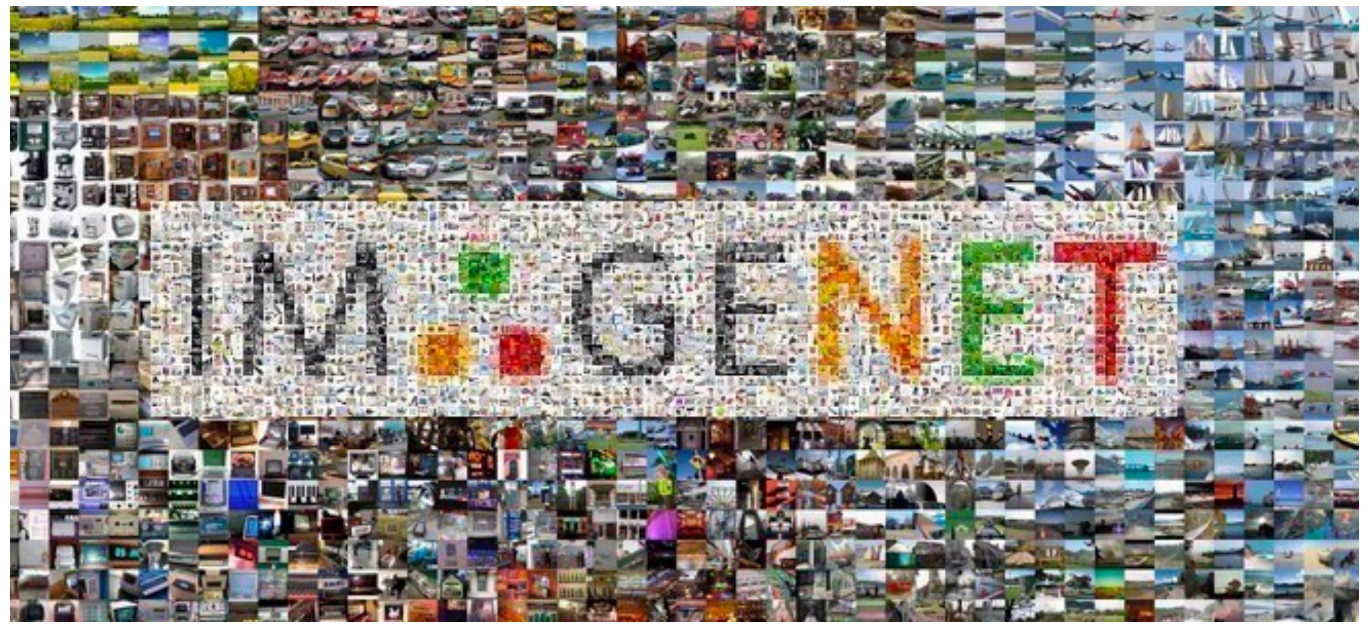
Turn-key solution for co-design



- Once-for-all model (ofa.mit.edu) sets a new state-of-the-art **80% ImageNet top-1 accuracy** under the mobile vision setting (< 600M MACs).

Applications

We focus on large-scale datasets to reflect real-life use cases.

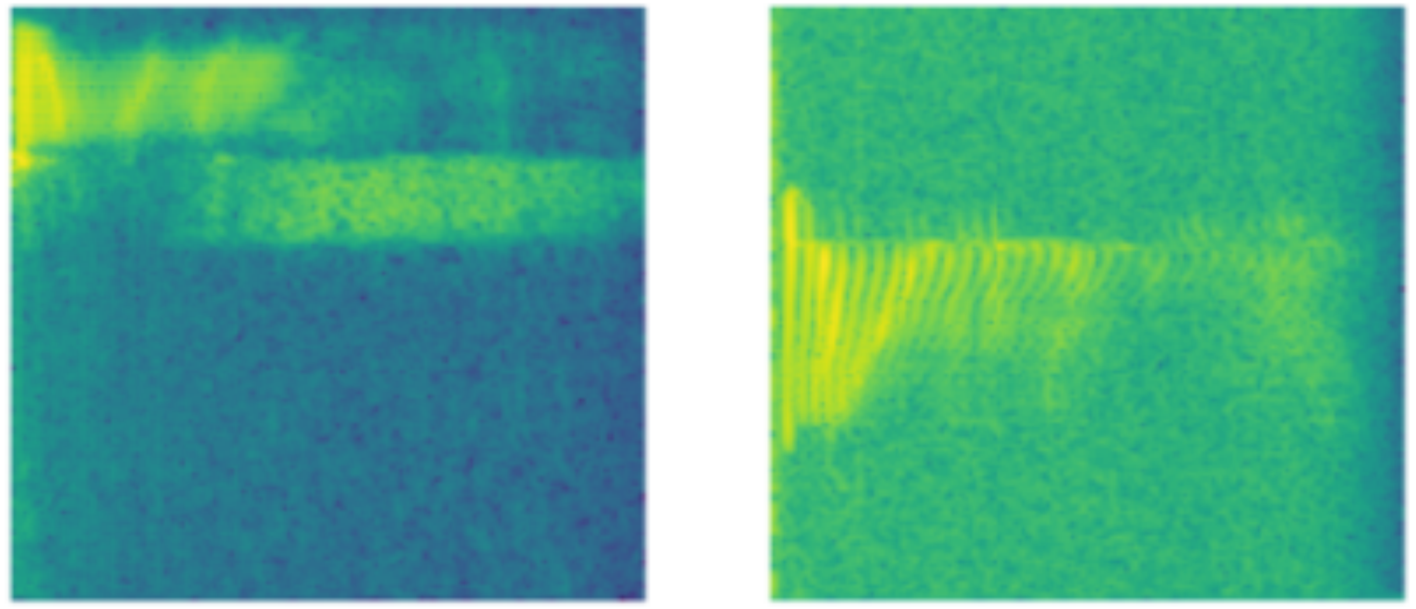


Datasets:

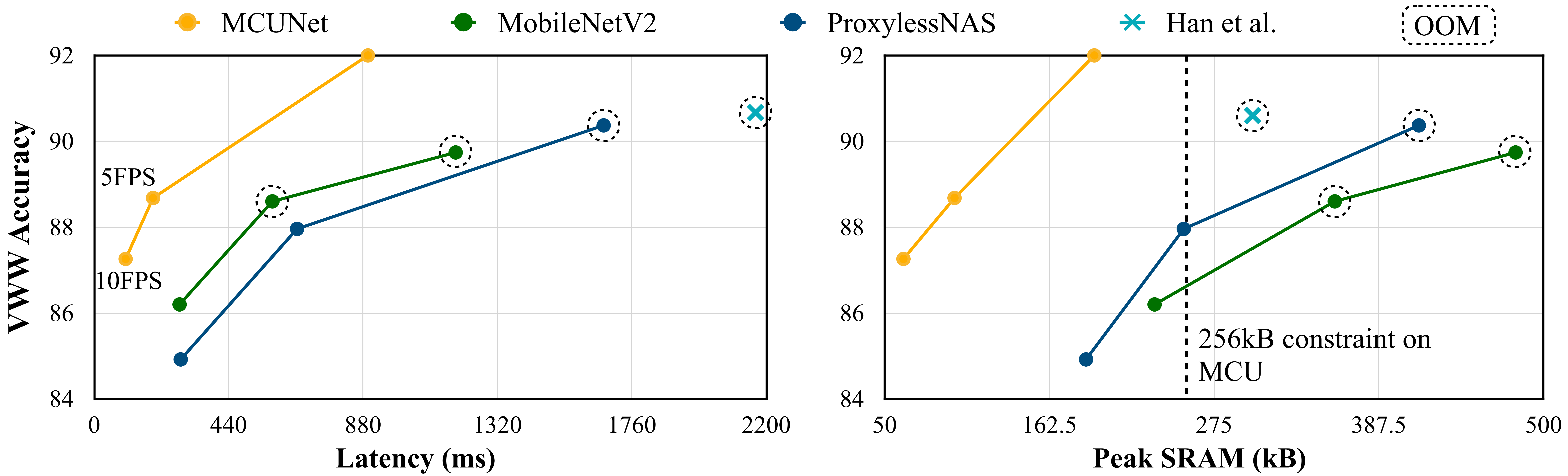
(1) ImageNet-1000
(2) Wake Words
- Visual: Visual Wake Words
- Audio: Google Speech Commands

(a) 'Person' (b) 'Not-person'

yes no



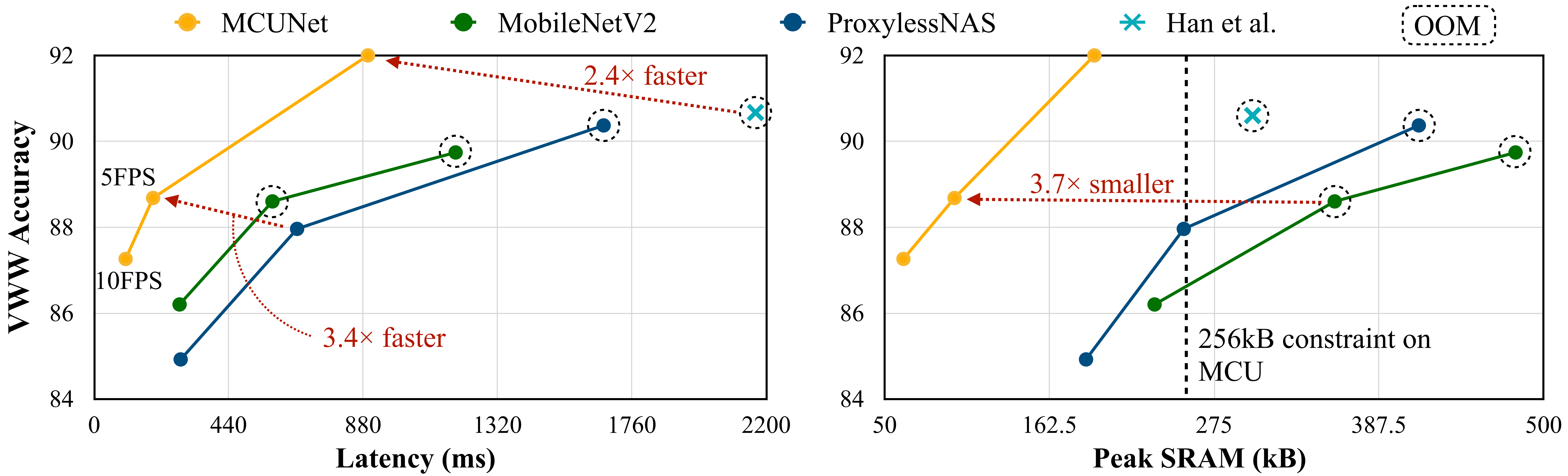
Visual Wake Words (VWW)



(a) Trade-off: accuracy *vs.* measured latency

(b) Trade-off: accuracy *vs.* peak memory

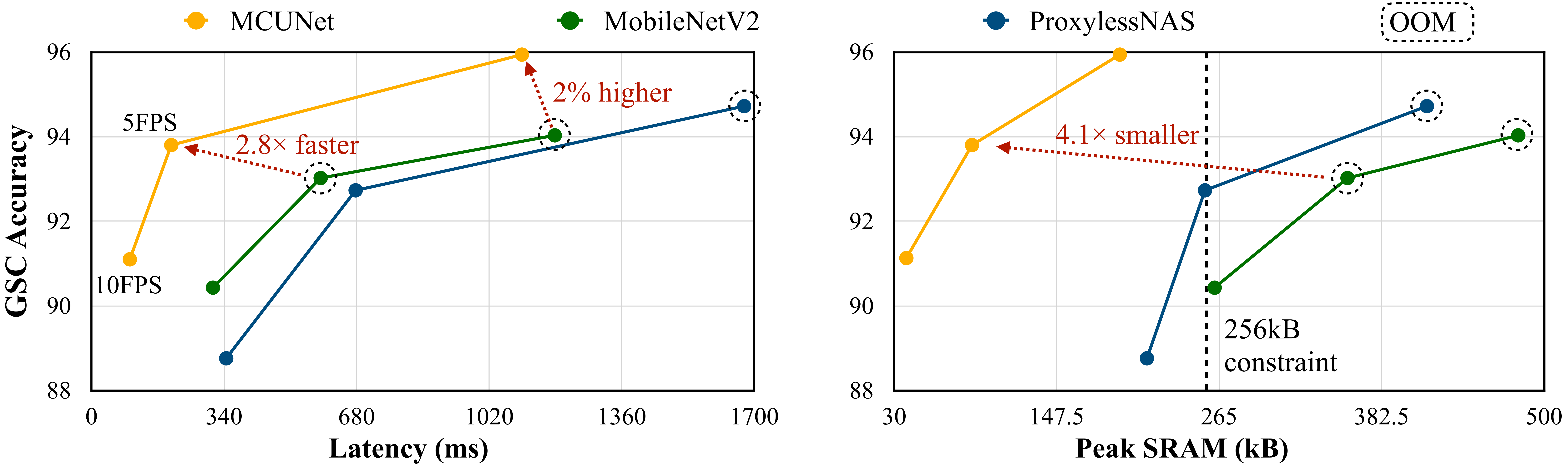
Visual Wake Words (VWW)



(a) Trade-off: accuracy *vs.* measured latency

(b) Trade-off: accuracy *vs.* peak memory

Audio Wake Words (Speech Commands)



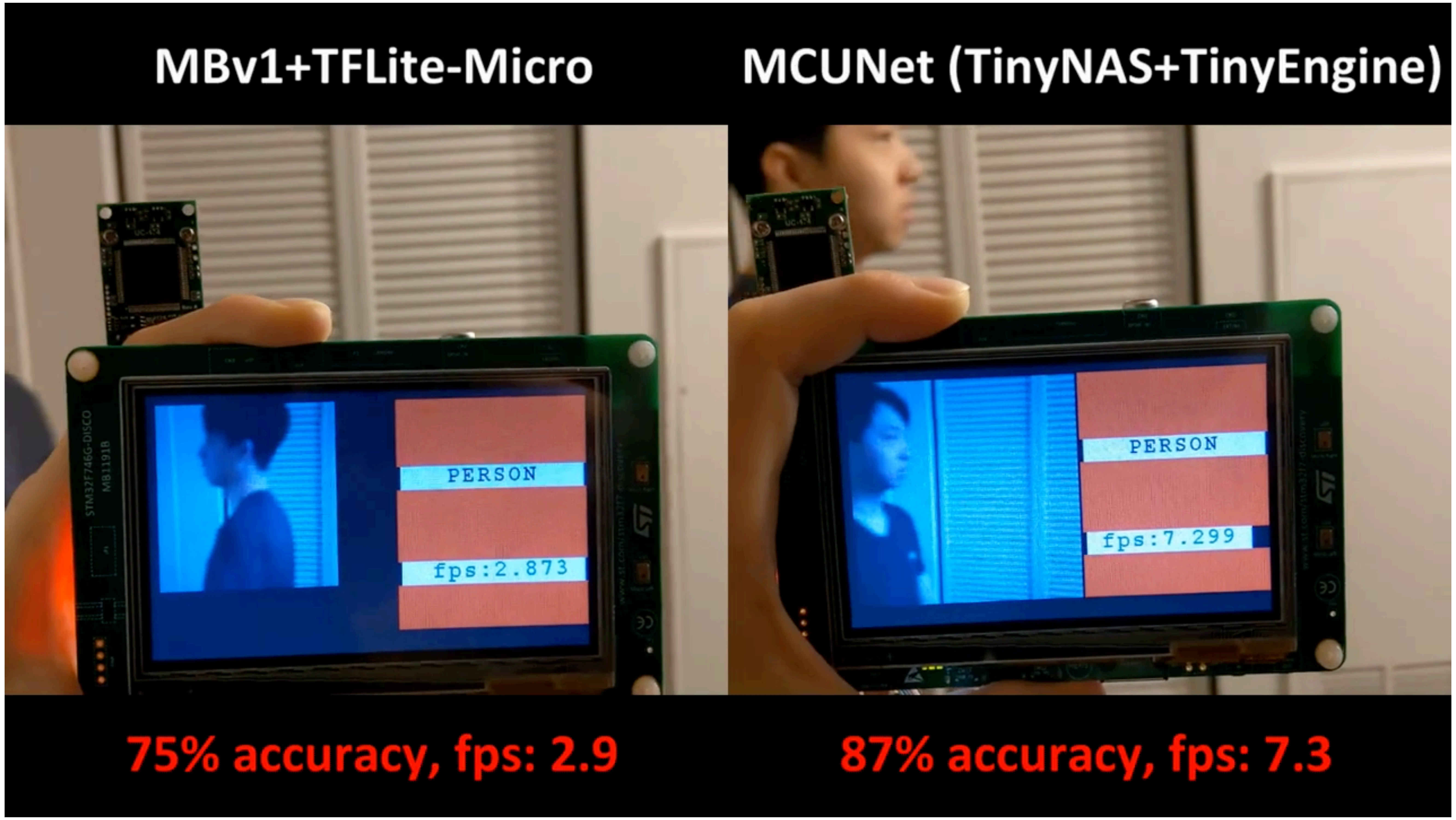
(a) Trade-off: accuracy *vs.* measured latency

(b) Trade-off: accuracy *vs.* peak memory

Visual Wake Word Detection

- Detecting whether a person is present in the frame

Demo:



UC-474
Rev A
P1 CAMERA
SPDIF IN
CN2
EXT/RF
RoHS
Compliant
MB1191C
STM32F746G-DISCO
U20
Micro Right
U21
Micro Left
www.st.com/stm32f7-discovery

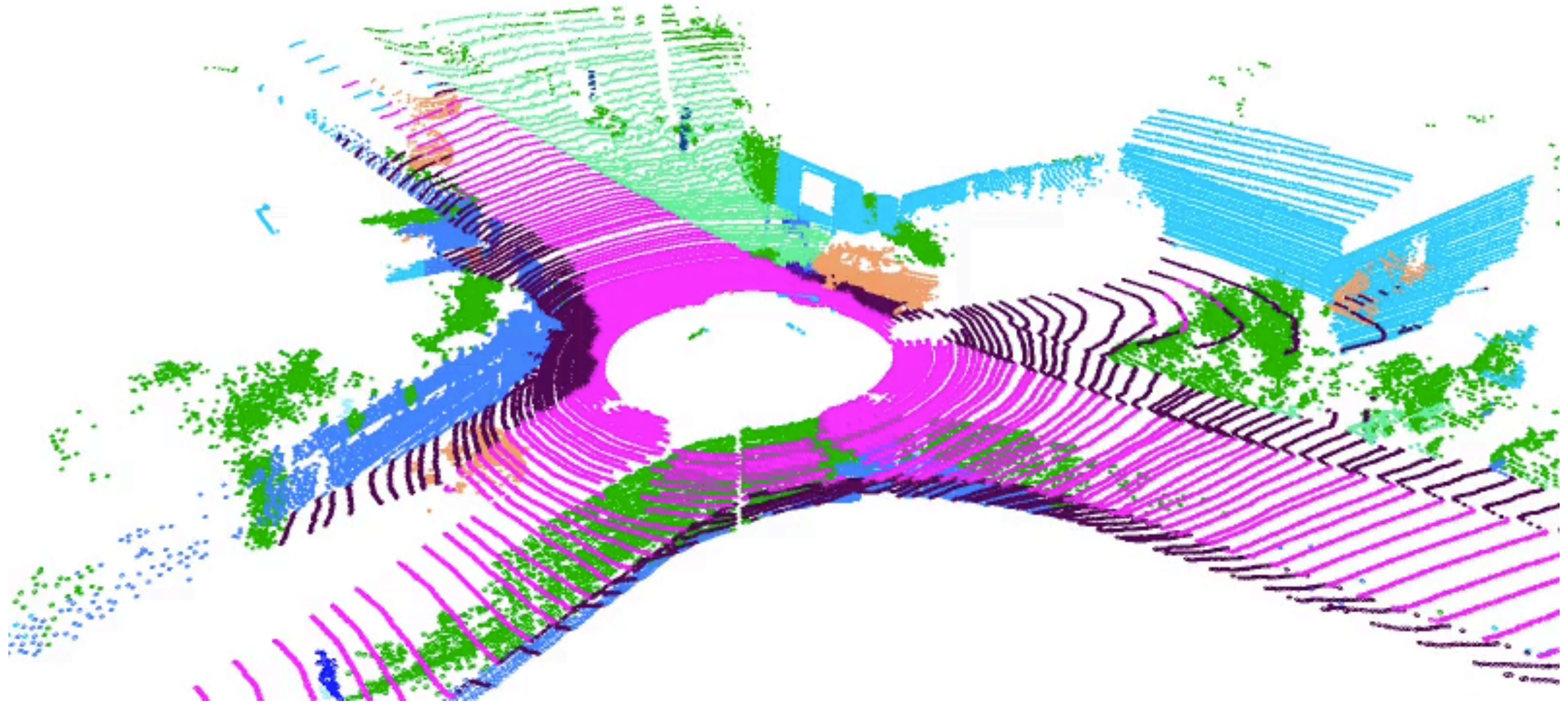
TinyML for Point Cloud

SPVNAS, ECCV'20

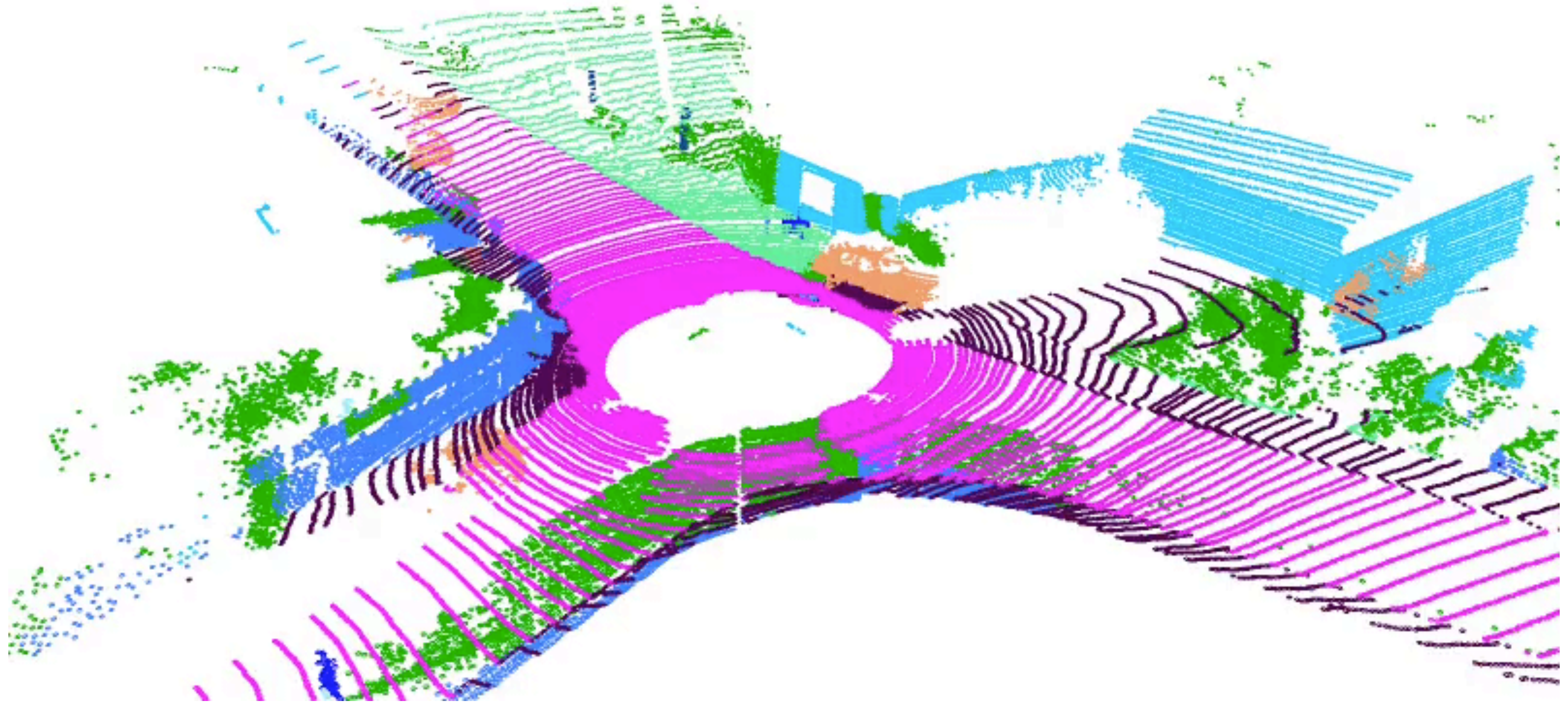
AR/VR: a whole backpack of computer

Self-driving: a whole trunk of GPU

Mobile phone: limited battery



MinkowskiNet: **3.4** FPS



SPVNAS (Ours): **9.1** FPS

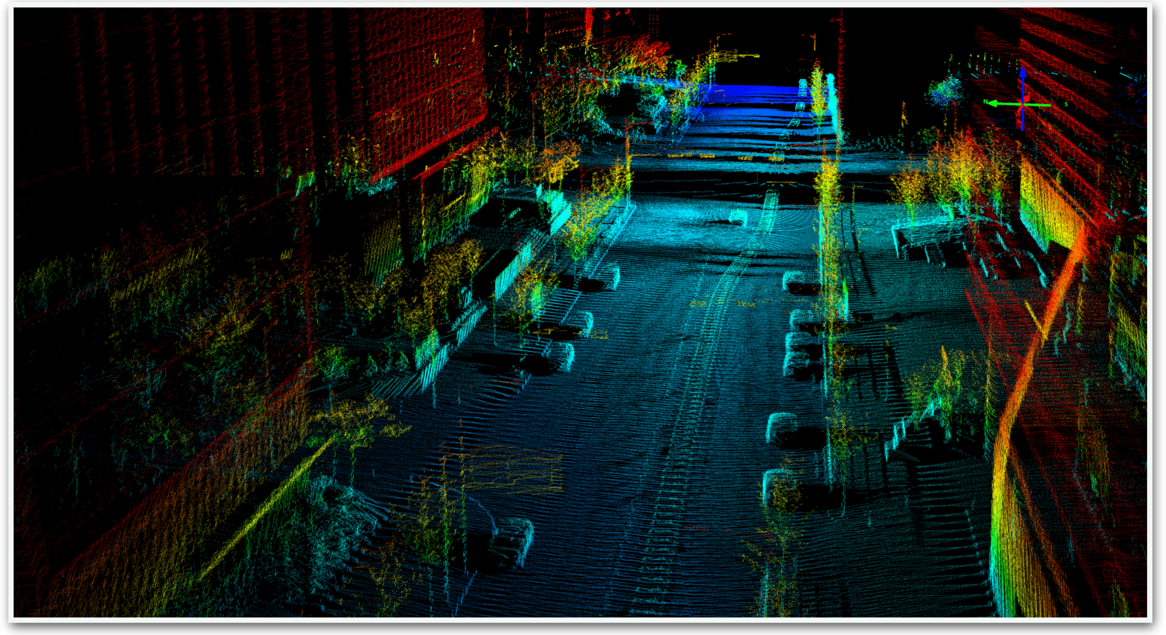
accuracy ranks 1st on the SemanticKitti leaderboard

Approach	Paper	Code	mIoU	Classes (IoU)
SPVNAS			67.0	
TORNADONet			63.1	
KPRNet			63.1	
Cylinder3D			61.8	
FusionNet			61.3	
SalsaNext			59.5	
KPConv			58.8	
SqueezeSegV3			55.9	

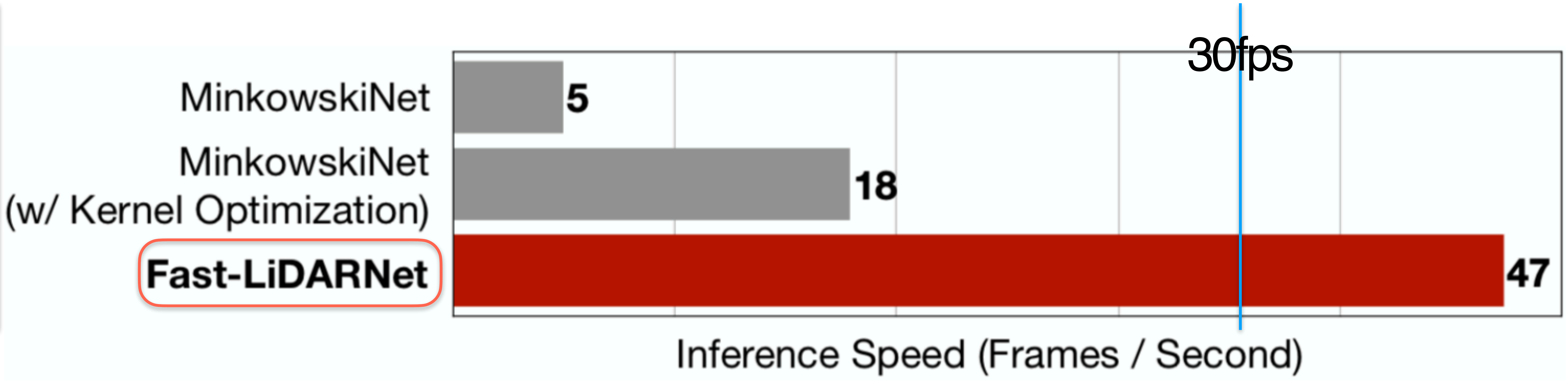
TinyML for Driving

Liu et al. ICRA'21.

3D LiDAR Sensor



3D Point Cloud: 2M points/s



Demo:

TinyML for GAN

Accelerating Horse2zebra by GAN Compression

Demo:

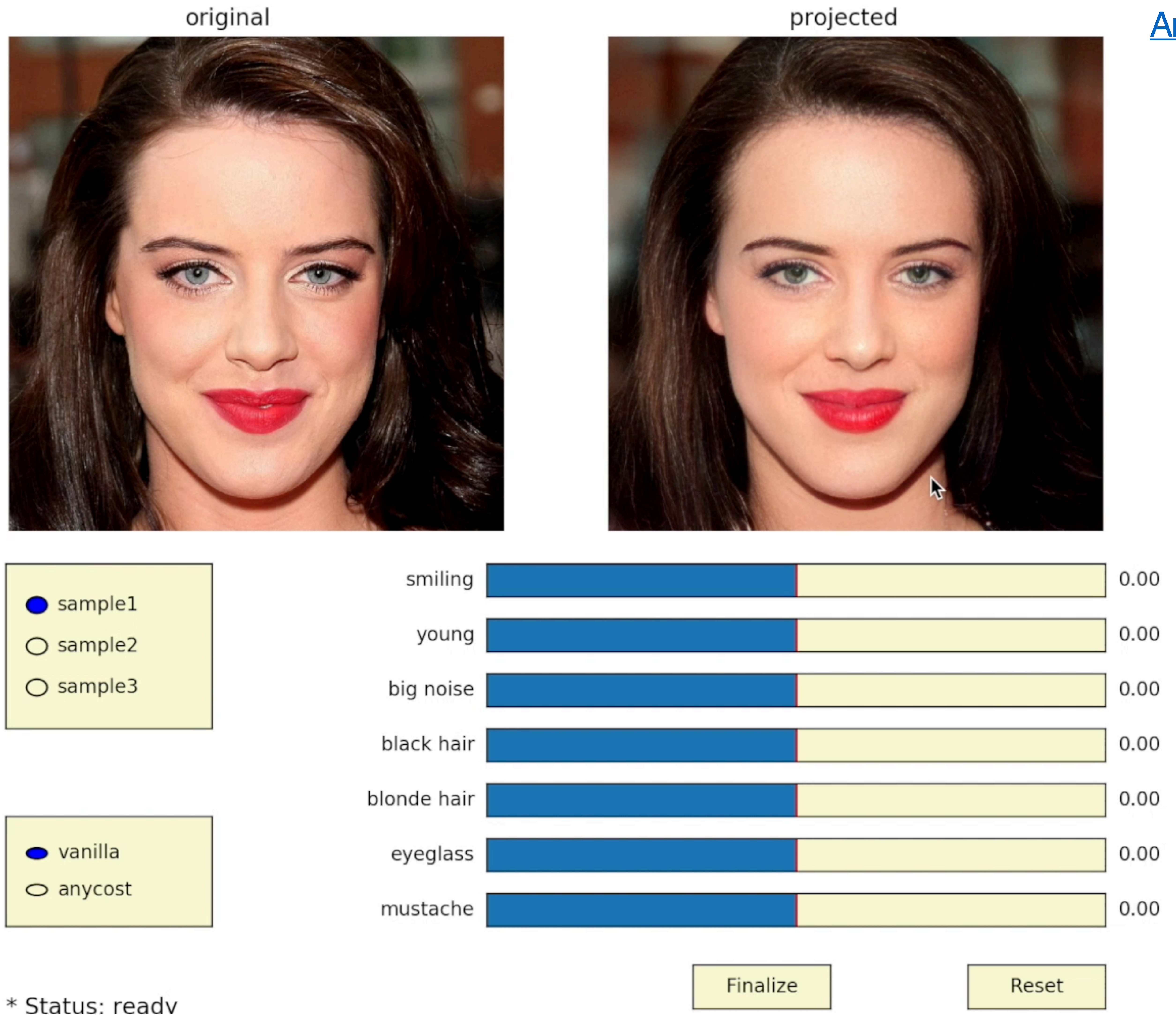
Original CycleGAN; FLOPs: 56.8G; FPS: 12.1; FID: 61.5

GAN Compression; FLOPs: 3.50G (16.2x); FPS: 40.0 (3.3x); FID: 53.6

TinyML for GANs

AnyCost GAN, CVPR'21

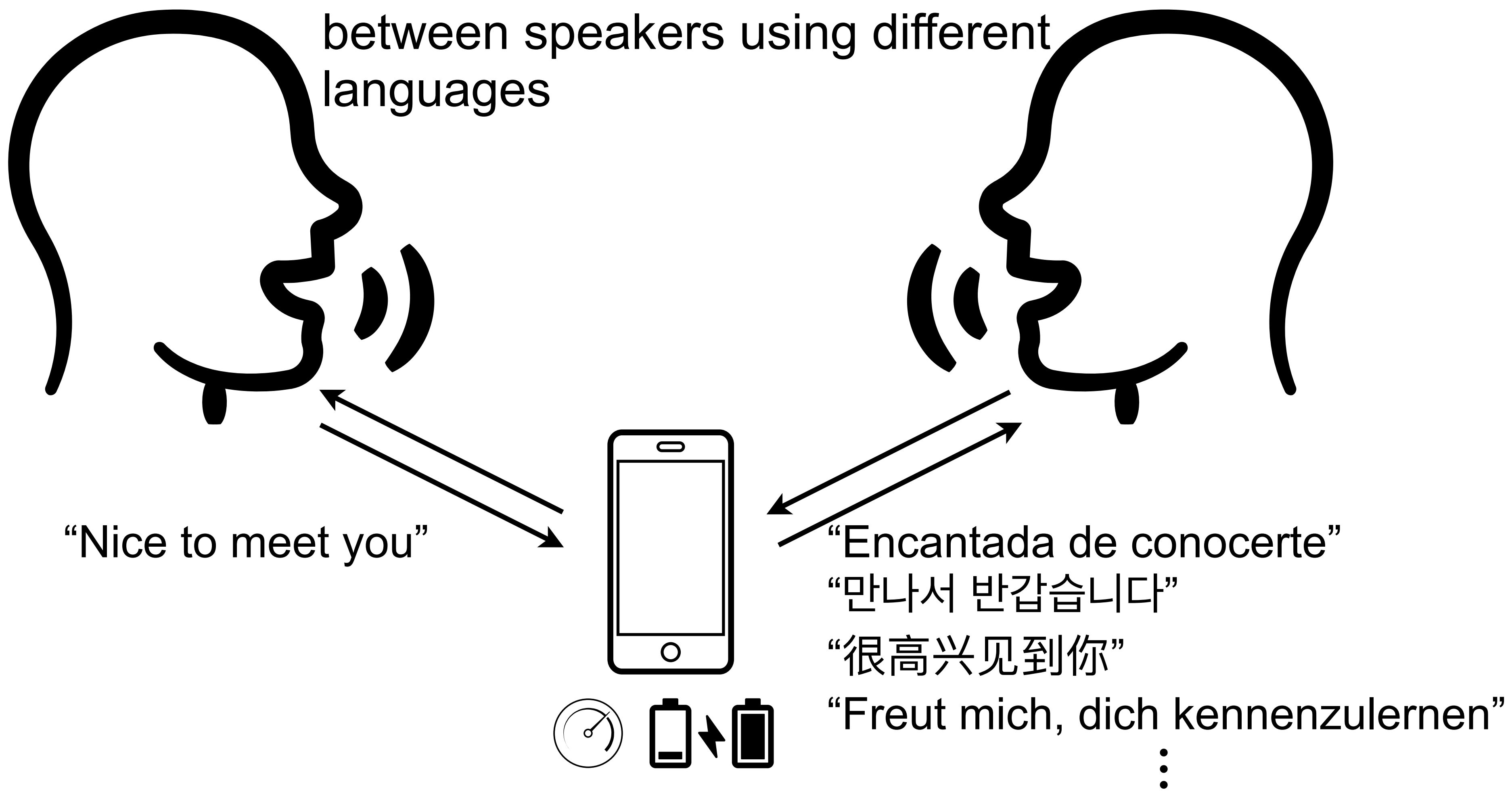
Demo:

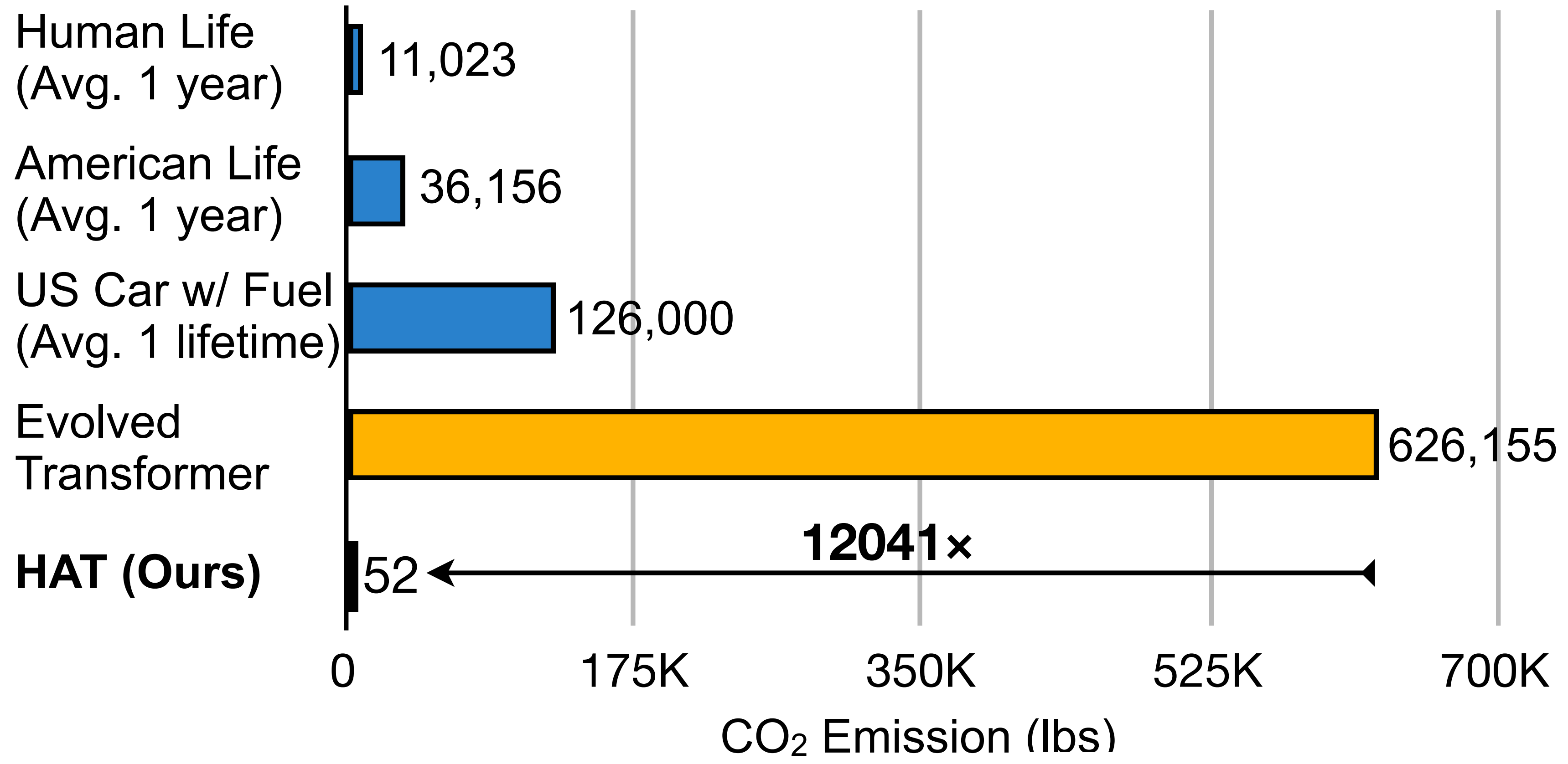


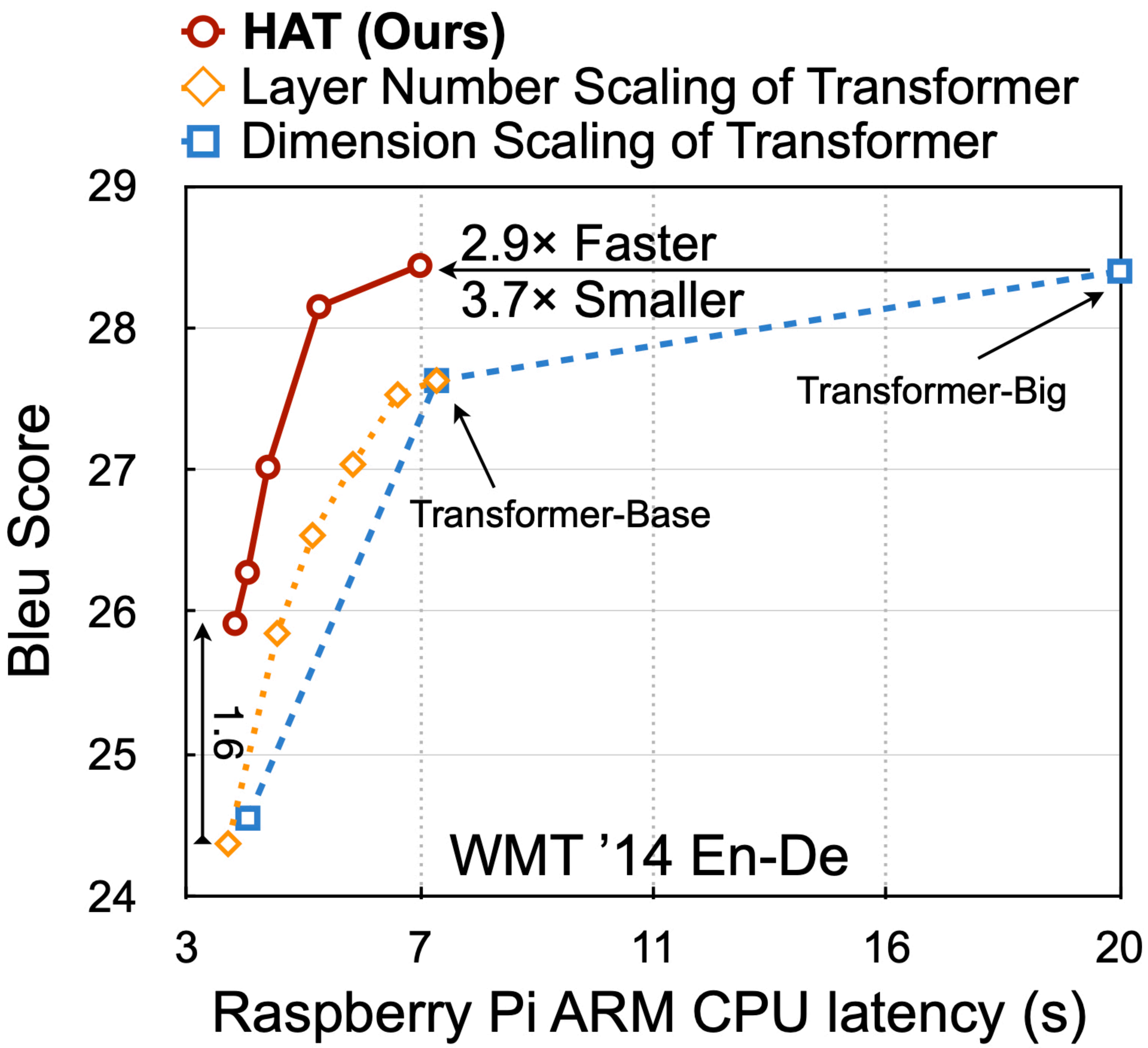
TinyML for NLP

HAT, ACL'20
SpAtten, HPCA'21

Efficient NLP on mobile devices enable real time conversation between speakers using different languages

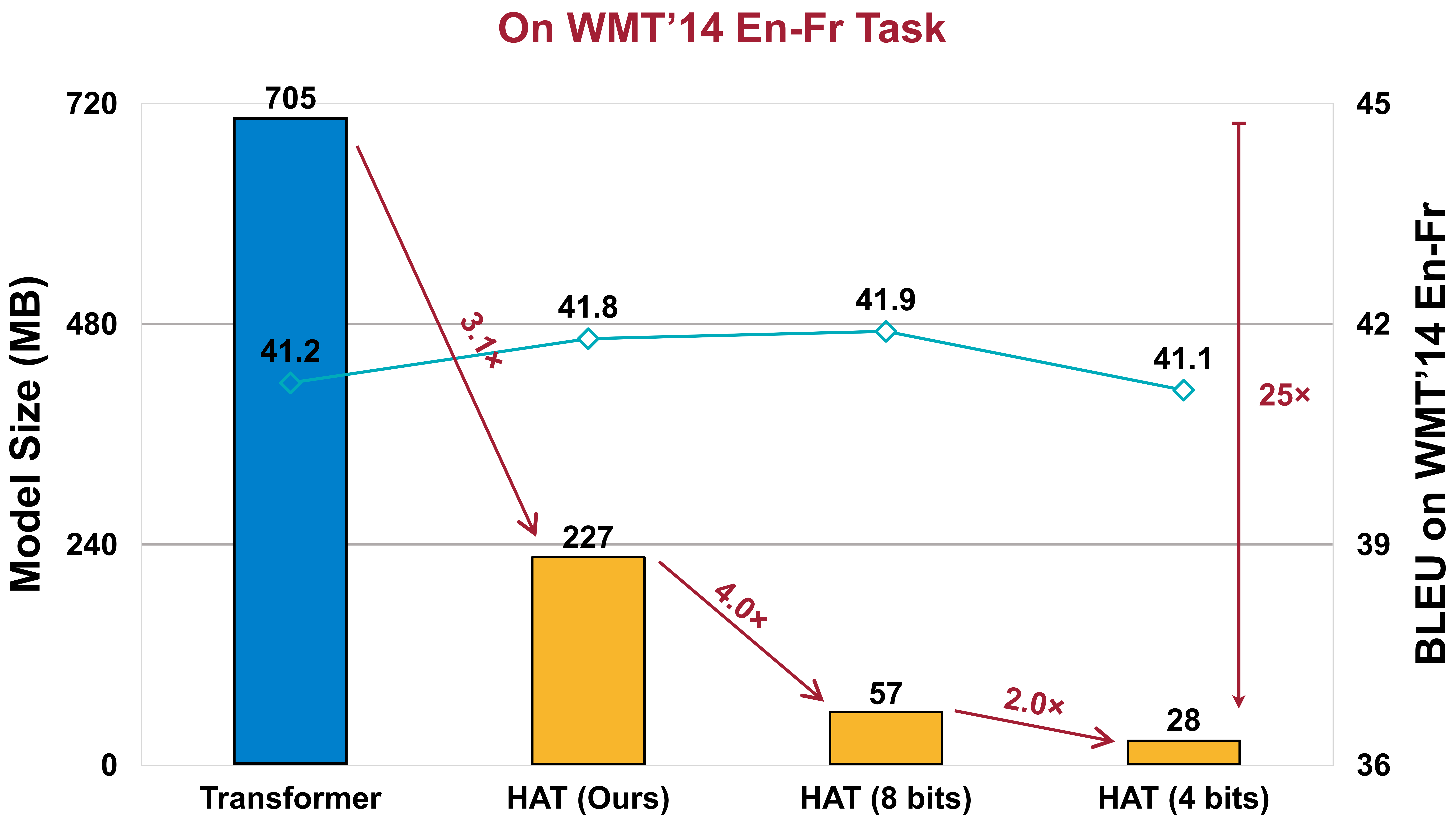






TinyML for NLP

HAT, ACL'20
SpAtten, HPCA'21



- HAT is orthogonal to general model compression techniques

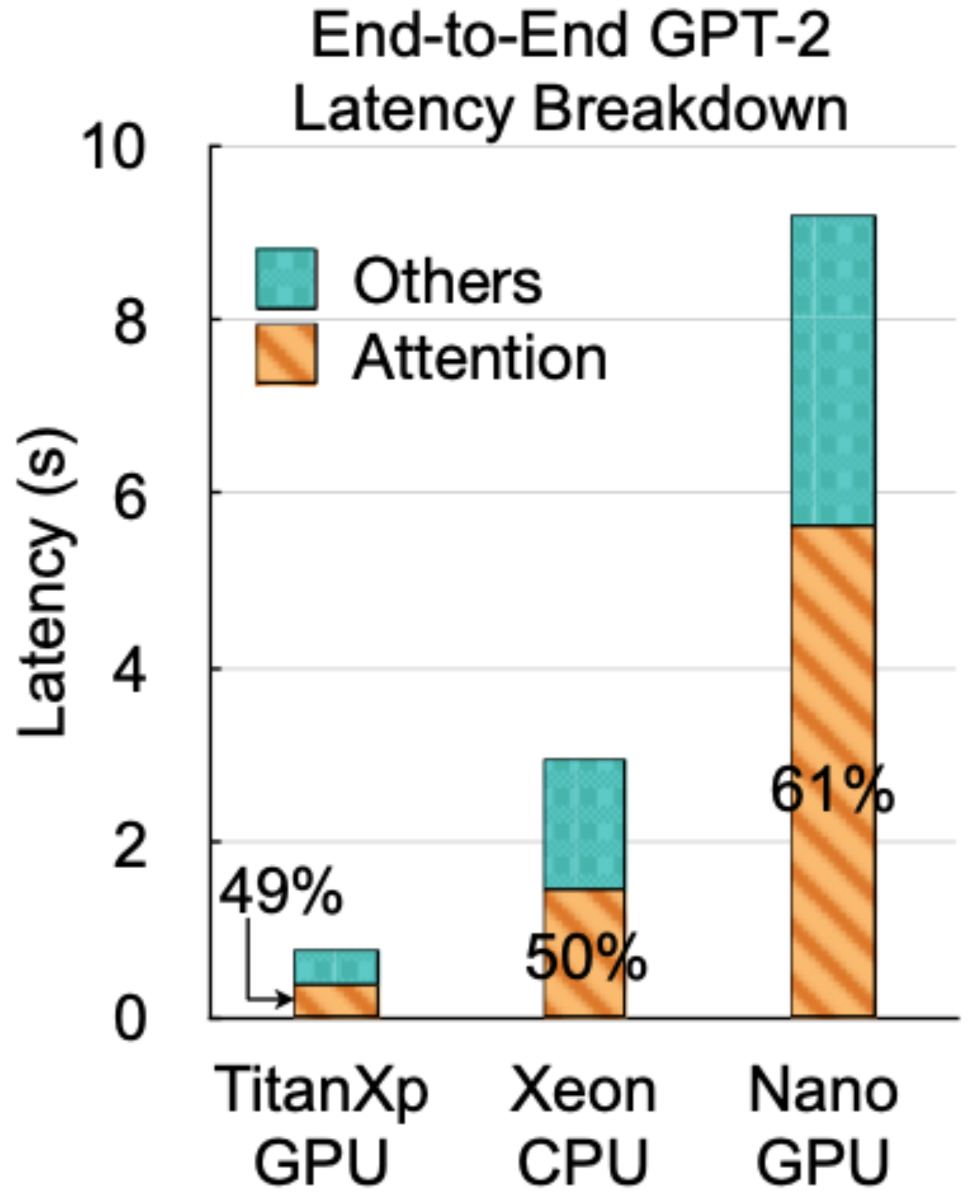
TinyML for NLP

HAT, ACL'20
SpAtten, HPCA'21

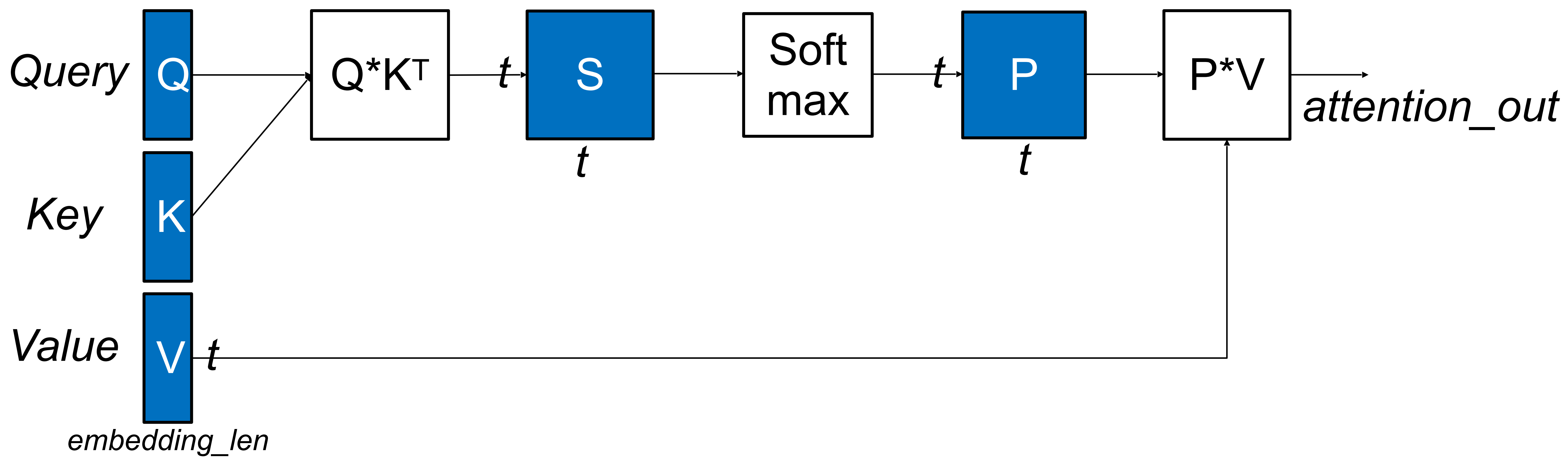
- **Motivation: Attention layer** in natural language processing models is the bottleneck for end-to-end performance.
- **Main idea: Reduce the redundant computation.**

1. **Cascade Token and head pruning**: Based on attention distribution, we **remove unimportant** tokens and heads to reduce computation and memory access.
2. **Progressive quantization:** progressively fetch MSB and LSB to **reduce average bitwidth**. If attention distribution is flat, using MSB is sufficient for accuracy.

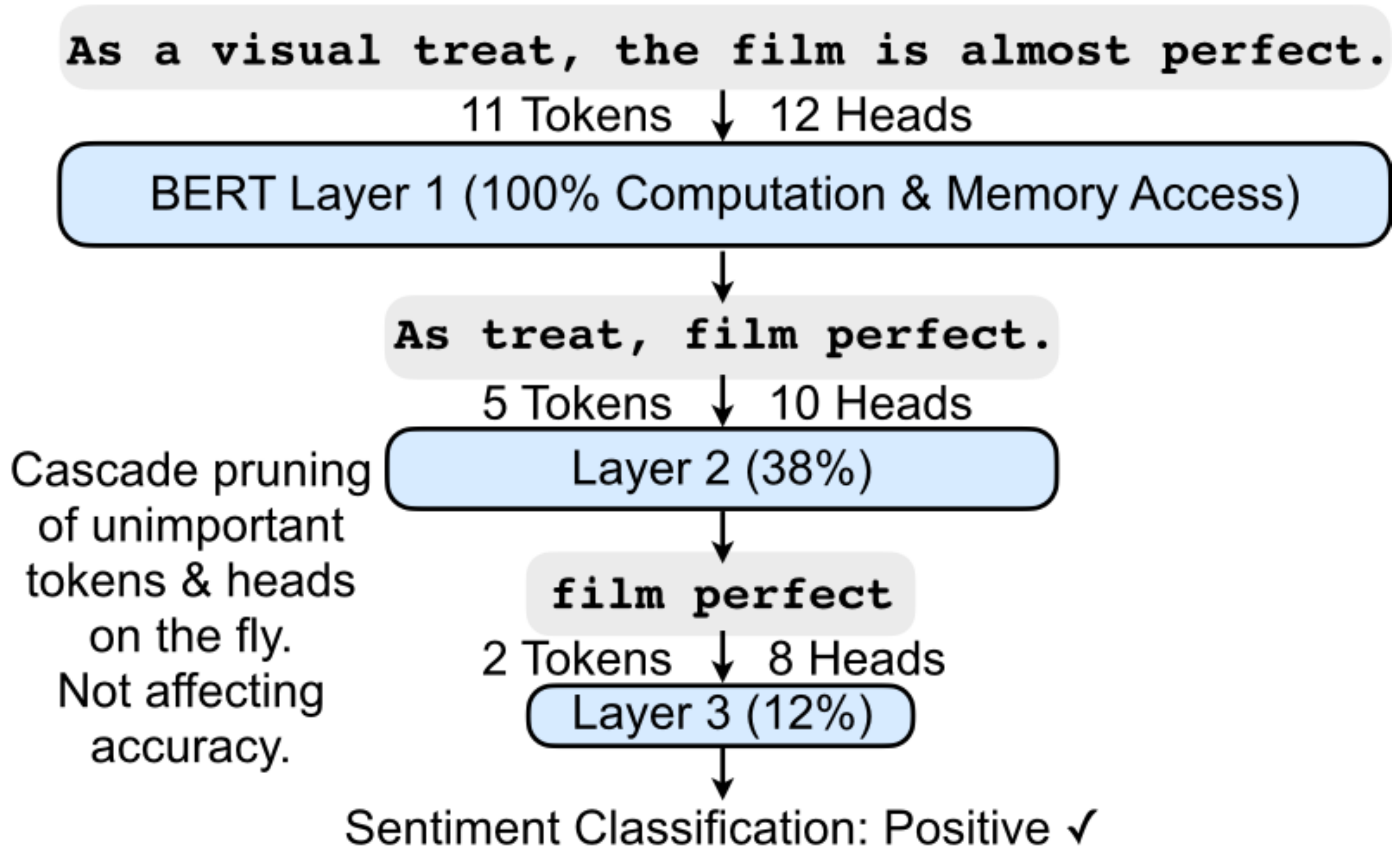
Attention operation is the bottleneck



Attention Mechanism



Cascade token and head pruning



[HPCA'21] Hanrui Wang, Zhekai Zhang, Song Han; "SpAtten: Efficient Sparse Attention Architecture with Cascade Token and Head Pruning"

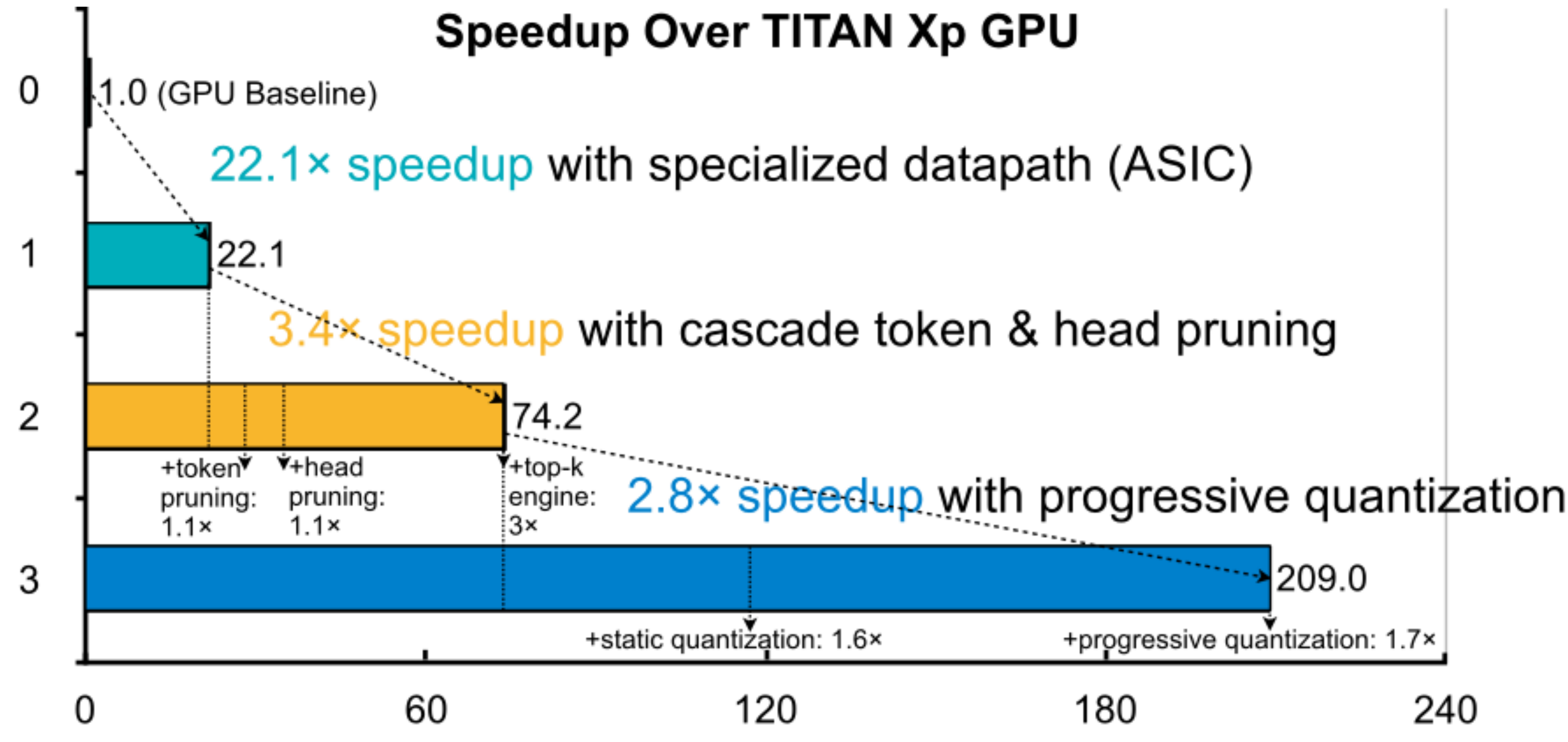
TinyML for NLP

Platform	Power (W)	Performance (GFLOPS)	Energy Efficiency (GFLOP/J)
Raspberry Pi (ARM)	3.49	0.33 (5,945x)	0.095 (2,529x)
Nvidia Nano (GPU)	2.88	1.58 (1,241x)	0.55 (457x)
Intel Xeon (CPU)	96.1	4.89 (401x)	0.051 (4,888x)
TITAN Xp (GPU)	56.7	10.6 (185x)	0.19 (1,428x)
SpAtten-full (ASIC)	7.96	1962	246

*SpAtten over general-purpose platforms

Platform	Power (W)	Performance (GFLOPS)	Energy Efficiency (GFLOP/J)
A3 (ASIC)	2.00	221 (1.6x)	269 (1.4x)
MNNFast (ASIC)	0.823	120 (3.0x)	120 (3.2x)
SpAtten-small (ASIC)	0.942	360	382

SpAtten* over state-of-the-art accelerators



Speedup breakdown of SpAtten over GPU

*SpAtten uses effective FLOPs, the pruned operations are considered executed

[HPCA'21] Hanrui Wang, Zhekai Zhang, Song Han; "SpAtten: Efficient Sparse Attention Architecture with Cascade Token and Head Pruning"

TinyML for Video Recognition

TSM, ICCV 2019

I3D:
Latency: **164.3** ms/Video   Something-V1 Acc.: **41.6**%

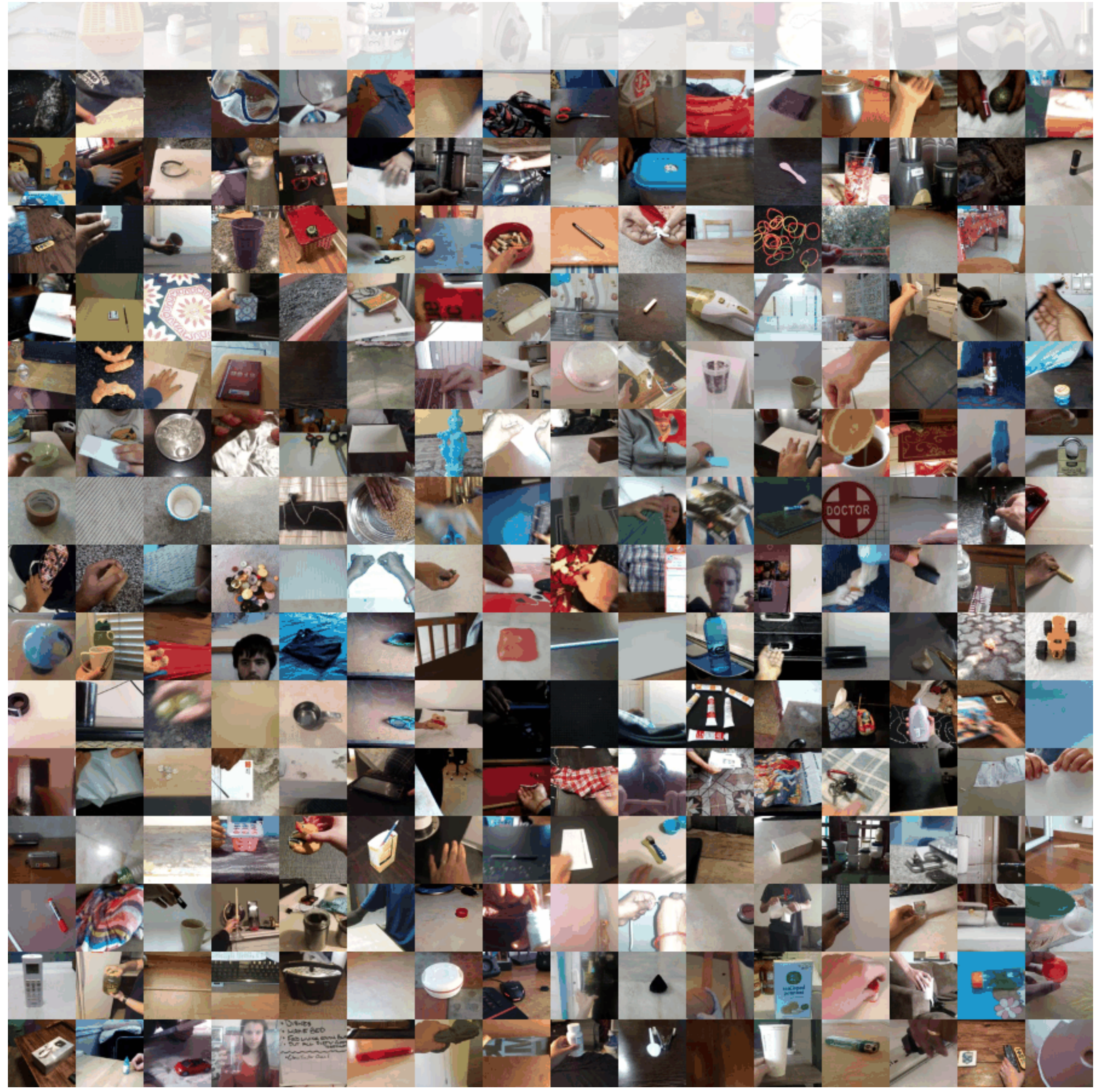
TSM:
Latency: **17.4** ms/Video   Something-V1 Acc.: **43.4**%

Speed-up: 9x

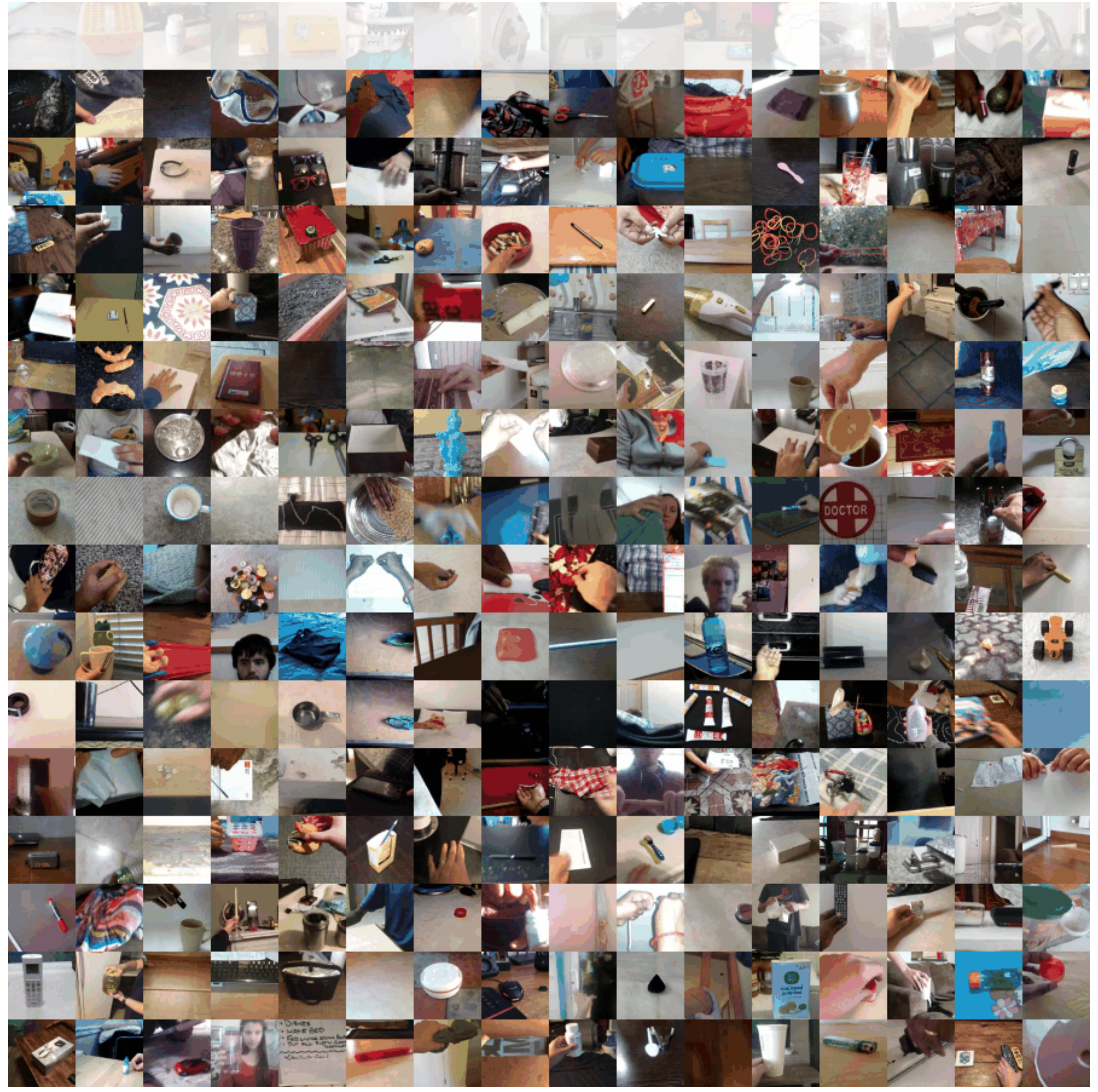
TinyML for Video Recognition

TSM, ICCV 2019

I3D:
Throughput: **6.1** video/s
Something-V1 Acc.: **41.6**%



TSM:
Throughput: **77.4** video/s
Something-V1 Acc.: **43.4**%



12.7x higher throughput

TinyML and Efficient Deep Learning

- **Optimize the Computation Efficiency**
  - Inference: MCUNet for IoT Devices [NeurIPS'20, spotlight]
  - Training: Tiny On-Device Transfer Learning (TinyTL) [NeurIPS'20]
- **Optimize the Data Efficiency**
  - Differentiable Augmentation for Data-Efficient GAN Training [NeurIPS'20]

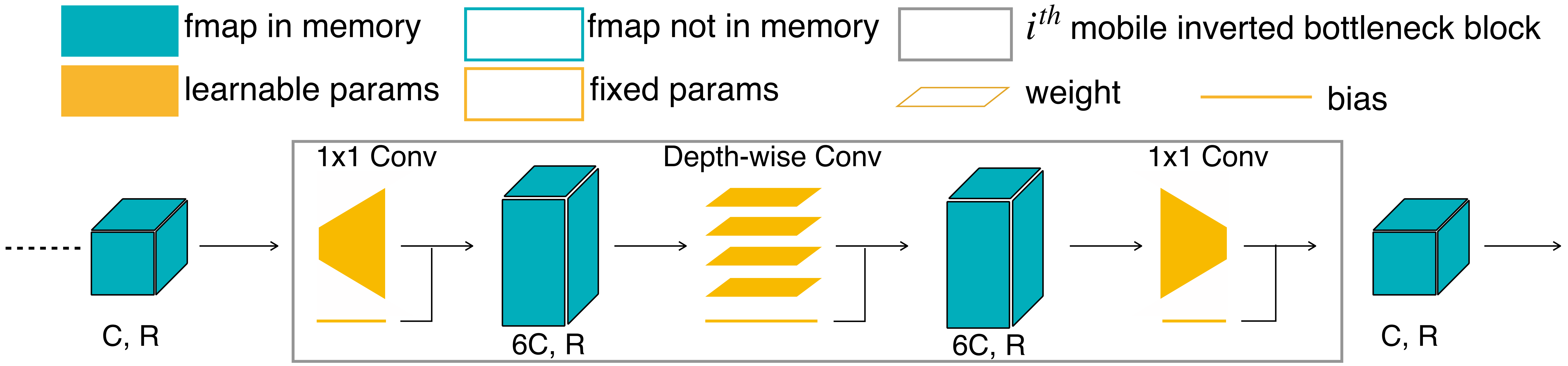
TinyTL: Reduce Memory, not Parameters for Efficient On-Device Learning

Han Cai[1] Chuang Gan[2] Ligeng Zhu[1] Song Han[1]

[1]MIT [2]MIT-IBM Watson AI Lab

NeurIPS 2020

Weight update is Memory-expensive; Bias update is Memory-efficient



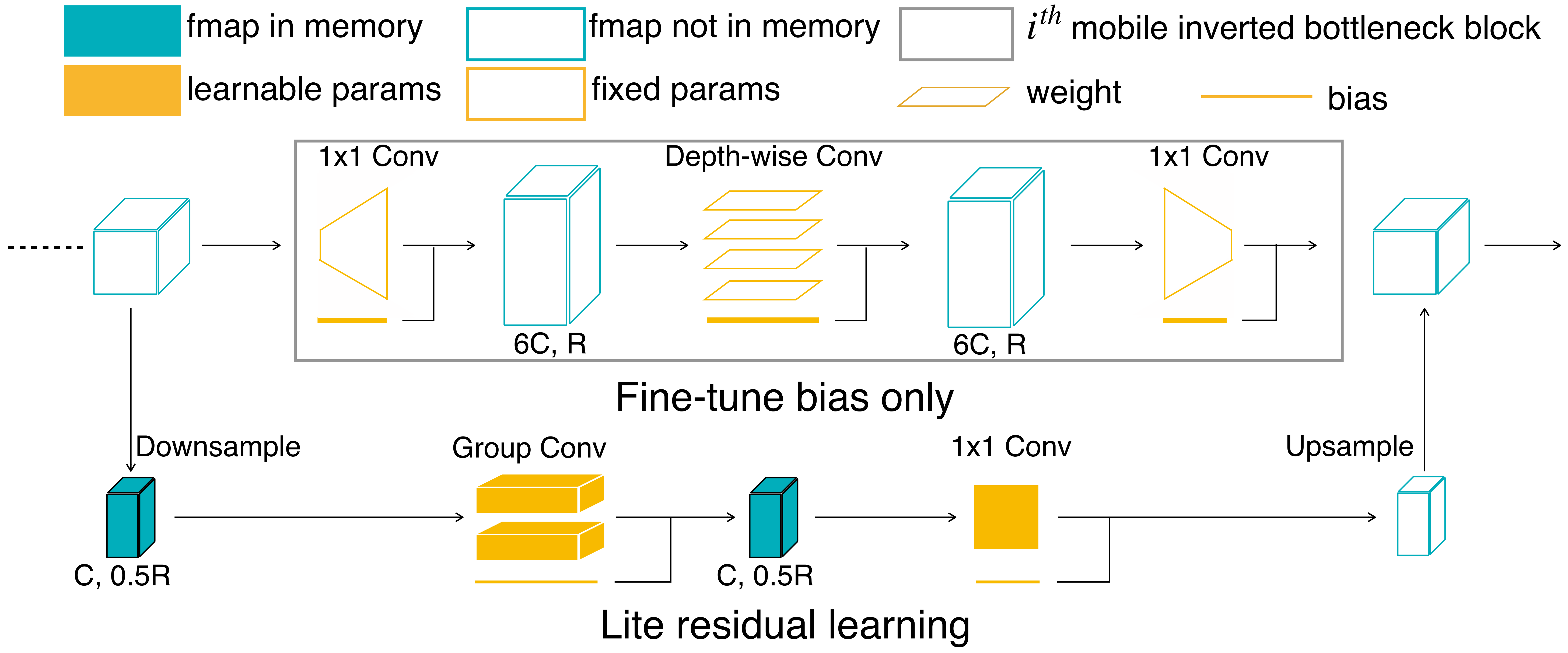
Fine-tune the full network (Conventional)

Forward: $\mathbf{a}_{i+1} = \mathbf{a}_i \mathbf{W}_i + \mathbf{b}_i$

Backward: $\dfrac{\partial L}{\partial \mathbf{W}_i} = \mathbf{a}_i^T \dfrac{\partial L}{\partial \mathbf{a}_{i+1}}, \qquad \dfrac{\partial L}{\partial \mathbf{b}_i} = \dfrac{\partial L}{\partial \mathbf{a}_{i+1}} = \dfrac{\partial L}{\partial \mathbf{a}_{i+2}} \mathbf{W}_{i+1}^T$

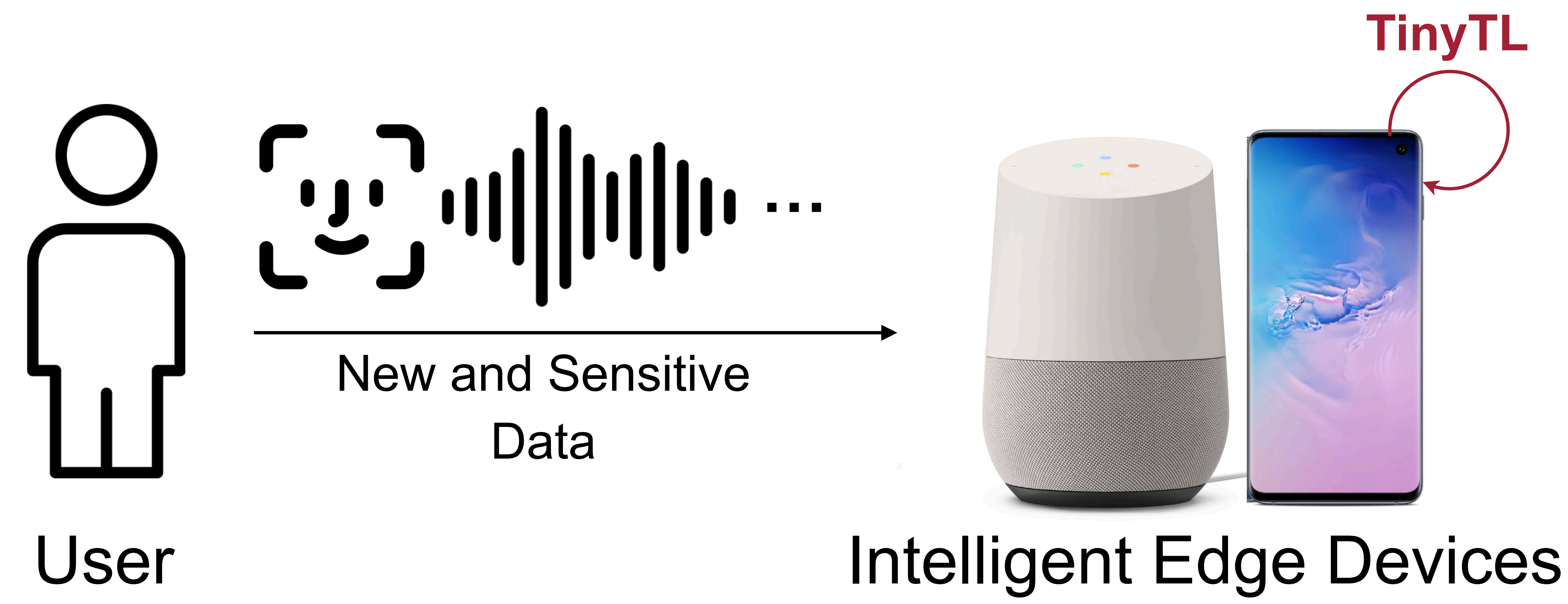
- Updating weights requires storing intermediate activations
- Updating biases does not

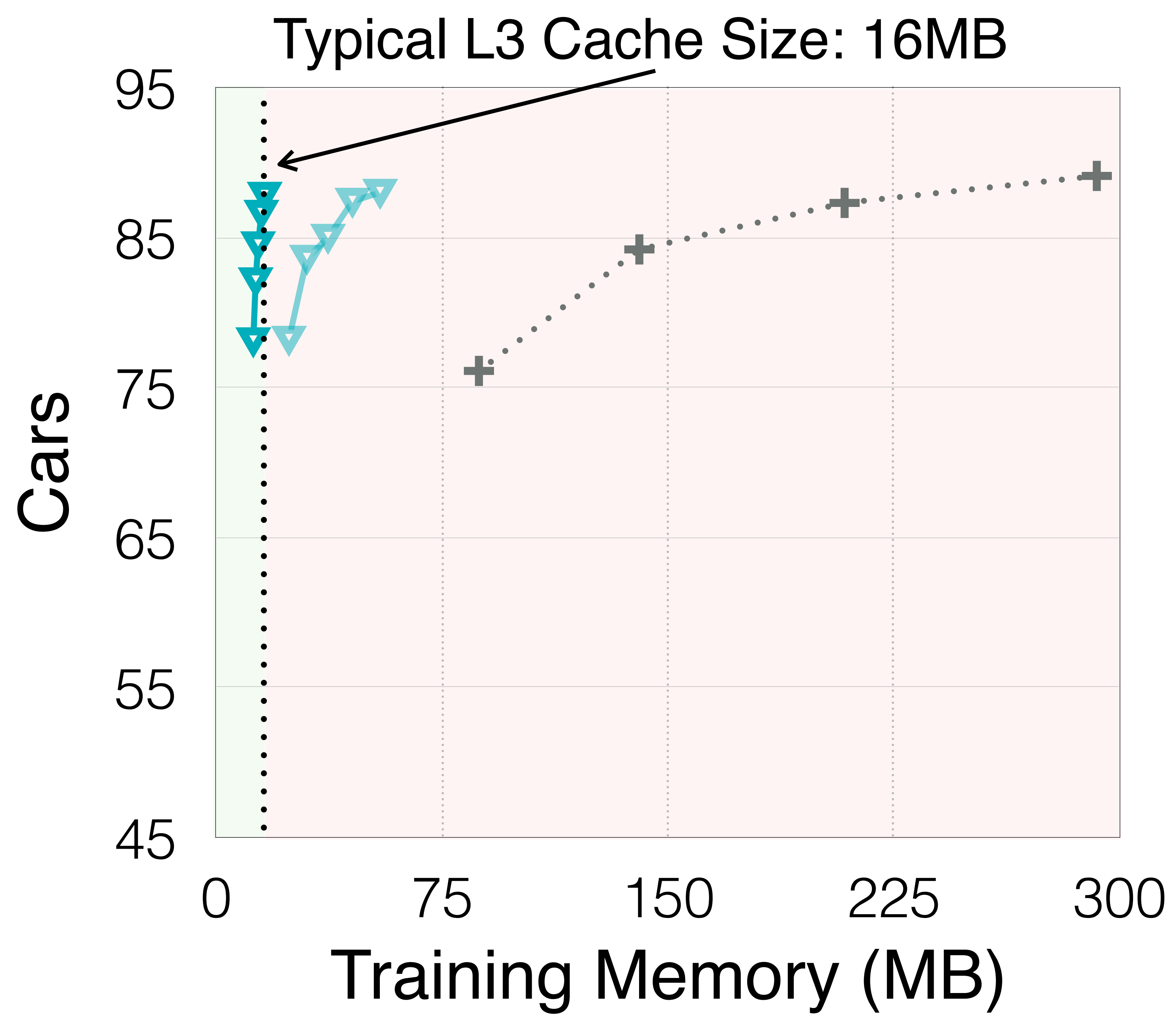
TinyTL: Lite Residual Learning



- Add lite residual modules (small memory overhead) to increase model capacity
  - (1/6 channel, 1/2 resolution, 2/3 depth)

TinyTL: Reduce Memory, not Parameters for Efficient On-Device Learning





Project Page: **http://tinyml.mit.edu**

TinyML and Efficient Deep Learning

- **Optimize the Computation Efficiency**
  - Inference: MCUNet for IoT Devices [NeurIPS’20, spotlight]
  - Training: Tiny On-Device Transfer Learning (TinyTL) [NeurIPS’20]
- **Optimize the Data Efficiency**
  - Differentiable Augmentation for Data-Efficient GAN Training [NeurIPS’20]

Differentiable Augmentation for Data-Efficient GAN Training

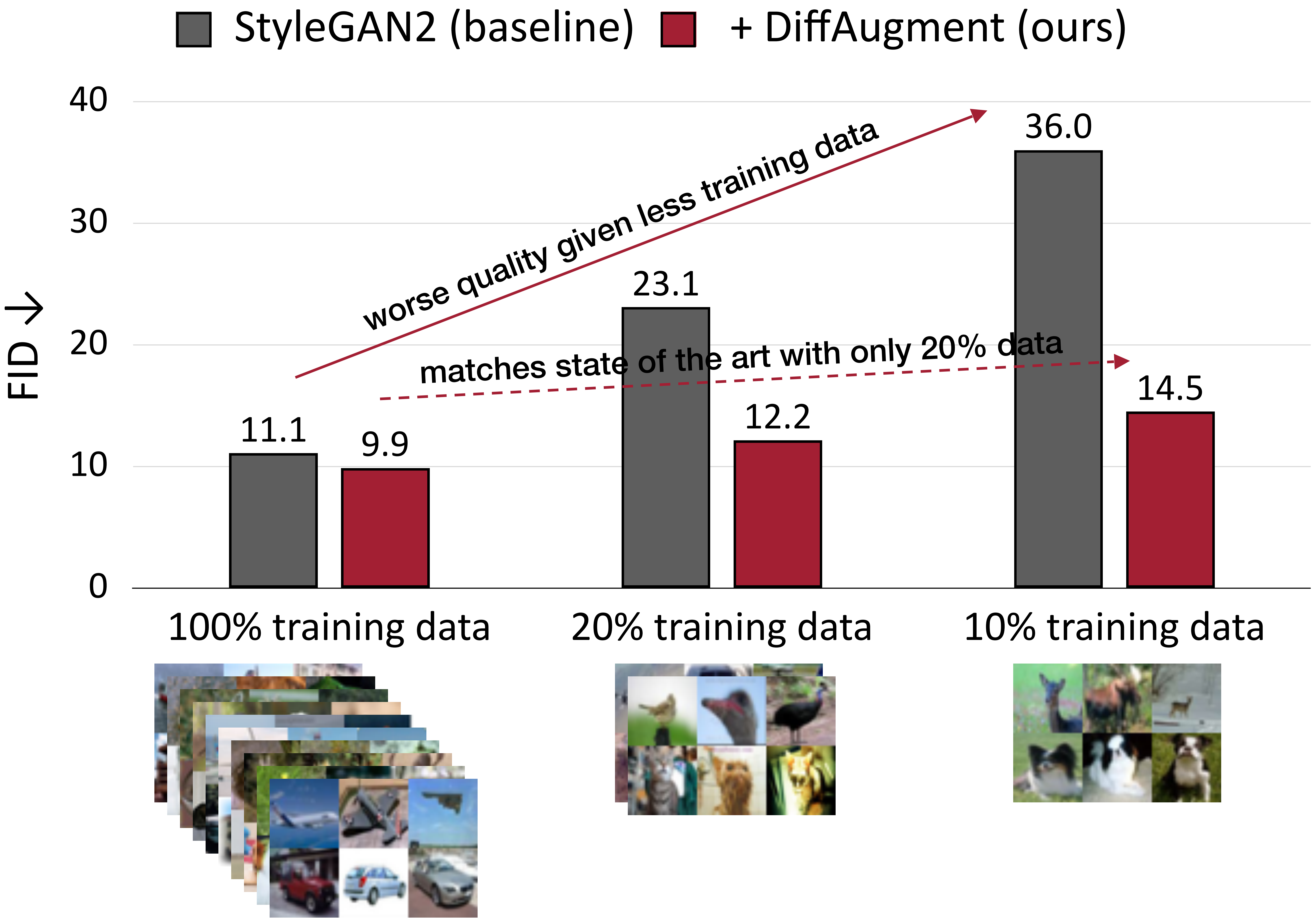
Shengyu Zhao[1,2] Zhijian Liu[1] Ji Lin[1] Jun-Yan Zhu[3,4] Song Han[1]

[1]MIT [2]IIIS, Tsinghua University [3]Adobe Research [4]CMU

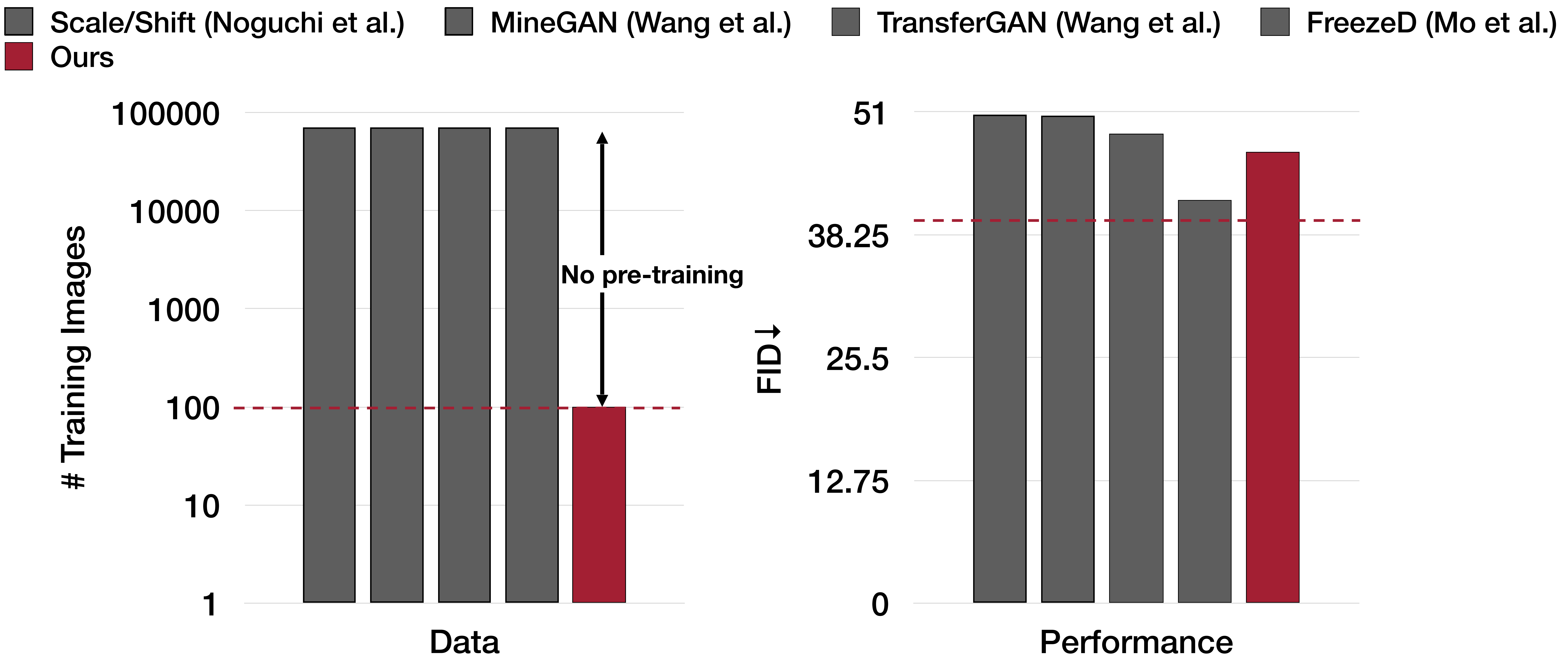
NeurIPS 2020

Low-Shot Generation

Our Results

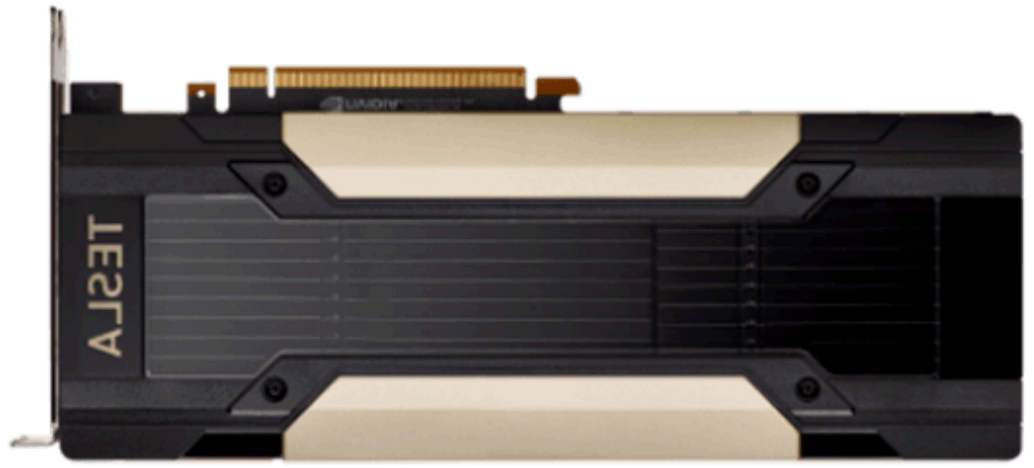

StyleGAN2 (baseline)
+ DiffAugment (ours)
FID ↓
40
30
20
10
0
worse quality given less training data
matches state of the art with only 20% data
11.1
9.9
23.1
12.2
36.0
14.5
100% training data
20% training data
10% training data

Fine-Tuning vs. Ours



100-shot Obama

Summary: TinyML and Efficient Deep Learning



Cloud AI	Mobile AI	Tiny AI
ResNet	MobileNet	**MCUNet**

- **Optimize the Computation Efficiency**
  - Inference: MCUNet for IoT Devices [NeurIPS'20, spotlight]
  - Training: Tiny On-Device Transfer Learning (TinyTL) [NeurIPS'20]
- **Optimize the Data Efficiency**
  - Differentiable Augmentation for Data-Efficient GAN Training [NeurIPS'20]

Project Page: **http://tinyml.mit.edu**

TinyML and Efficient AI

MIT HAN LAB
Hardware, AI and Neural-nets

github.com/mit-han-lab

youtube.com/c/MITHANLab

songhan.mit.edu